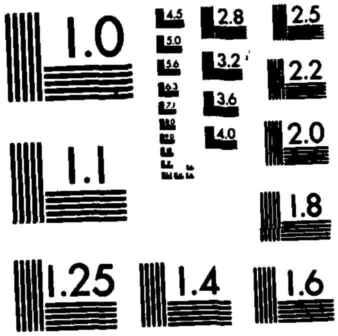
AD-A124 781 ENHANCED IMAGE TRACKING: ANALYSIS OF TWO ACCELERATION 1/6
MODELS IN TRACKING..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. M R KOZEMCHAK
UNCLASSIFIED DEC 82 AFIT/GEO/EE/82D-4 F/G 17/7 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD A124781

DTIC
ELECTE
FEB 23 1983
S E D

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY (ATC)

AIR FORCE INSTITUTE OF TECHNOLOGY

DTIC FILE COPY

Wright-Patterson Air Force Base, Ohio

This document has been approved for public release and sale; its distribution is unlimited.

83 02 022 0

AFIT/GEO/EE/82D-4

1

ENHANCED IMAGE TRACKING: ANALYSIS OF TWO ACCELERATION MODELS IN TRACKING MULTIPLE HOT-SPOT IMAGES

THESIS

AFIT/GEO/EE/82D-4   MARK R. KOZEMCHAK
2d LT   USAF

DTIC
S ELECTE D
FEB 23 1983
E

Approved for public release; distribution unlimited.

AFIT/GEO/EE/82D-4

ENHANCED IMAGE TRACKING:
ANALYSIS OF TWO ACCELERATION MODELS IN TRACKING MULTIPLE HOT-SPOT IMAGES

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Electrical Engineering

by

Mark R. Kozemchak B.S.E.E.
2d LT USAF
Graduate Electro-Optics
December 1981

Preface

This study was part of an ongoing effort to design an effective tracking algorithm for use with one of the Air Force Weapons Laboratory's high energy laser weapons. The algorithm uses modern optimal estimation and control techniques in the design of extended Kalman filters that process the outputs of forward looking infrared sensors.

I wish to express my sincerest thanks to my advisor, Dr. Peter S. Maybeck, for his motivation, expert advice, and valuable time. His comments were always helpful in keeping this research on track. Additional thanks are due to Lt. Holly L. Emrick, a very special friend, for her encouragement and help on many of the details of compiling this report.

I also wish to thank my family, especially my parents, for their love, concern, and never ending support of all my efforts, whatever they may be. Finally, this author would like to acknowledge the entire Graduate Electro-Optics class of December 1982 for their friendship through all that we have experienced in the past one and a half years.

Accession For
NTIS GRA&I ☒
DTIC TAB ☐
Unannounced ☐
Justification
By
Distribution/
Availability Codes

Dist	Avail and/or Special
A	

TIC COPY INSPECTED

Contents

Page

Preface ii

List of Figures v

List of Tables viii

List of Symbols ix

Abstract xiv

I. Introduction 1

1.1 Background 2
1.2 Problem 6
1.3 Assumptions 12
1.4 Overview 13

II. Derivation of Intensity Functions 15

2.1 Introduction 15
2.2 Two-Dimensional Fourier Transform 16
2.3 Exponential Smoothing 19
2.4 Shifting Property 21
2.5 Derivative property 24
2.6 Summary 25

III. Truth Model Development 28

3.1 Introduction 28
3.2 Atmospheric Disturbance Model 29
3.3 Target Dynamics Model - Trajectory Generation 34
3.4 Overall State Space Model 55
3.5 Measurements 58
3.6 Target Image 60
3.7 Spatially Correlated Background Noise 66
3.8 Summary 72

IV. Extended Kalman Filter Models 74

4.1 Introduction 74
4.2 First Order Gauss-Markov Acceleration Model 75
4.3 State Propagation of the Gauss-Markov Model 78
4.4 Constant Turn-Rate Acceleration Model . . . 80
4.5 State Propagation of the Constant Turn-Rate Model 83

Contents

Page

4.6 Measurement Update Equations 85
4.7 Dynamic Driving Noise Estimation 88

V. Performance Analysis Methodology 92

5.1 Introduction 92
5.2 Monte Carlo Analysis 92
5.3 Figures of Merit 94
5.4 Variation of Parameters 99
5.5 Summary 107

VI. Results 108

6.1 Introduction 109
6.2 Effect of Modelling Spatial Correlations . .
6.3 Basic Filter Comparisons 114
 6.3.1 Single Hot-Spot Image 114
 6.3.2 Three Hot-Spot Image 118
6.4 Performance vs Changing Target Dynamics - Dynamic Three Hot-Spot image 122
 6.4.1 Constant G Maneuvers 123
 6.4.2 Constant Roll-Rate Maneuvers 127
6.5 Dynamic Driving Noise Estimation 131

VII. Conclusions and Recommendations 133

7.1 Conclusions 133
7.2 Recommendations 135

Bibliography 138

Appendix A: Derivation of Truth Model $\underline{Q}_{TD}$ Matrix . . . 140

Appendix B: Derivation of Constant Turn-Rate Filter $\underline{F}$ and $\underline{\Phi}$ Matrices 144

Appendix C: Generation of a White Gaussian Noise Process 148

Appendix D: Performance Plots of Gauss-Markov Filter 150

Appendix E: Performance Plots of Constant Turn-Rate Filter 267

Appendix F: Performance Plots of Brownian Motion Filter 405

Appendix G: Computer Software 446

List of Figures

Figure		Page
1	Data Processing Algorithm	9
2	Third Order Shaping Filter	29
3	Inertial Reference Frame	37
4	Azimuth Geometry	38
5	Elevation Geometry	39
6	Target and $\alpha\beta$ Planes	44
7	Trajectory 1	46
8	Trajectory 2	48
9	Trajectory 3	49
10	Target Geometry	52
11	FLIR Image Geometry	61
12	Plane Perpendicular to the LOS	63
13	Pixel Numbering Scheme	68
14	Non-zero Correlations with the k-th Pixel . .	69
15	Generation of Error Statistics	93
16	Sample Plot - Mean Error +/- Standard Deviation	97
17	Sample Plot - Filter Sigma vs True RMS error	98
18	Static Multiple Hot-Spot Target Image	102
19	Dynamic Multiple Hot-Spot Target Image . . .	102

List of Figures

Figures **Pages**

D-1 to D-14 Case 1 GM Performance Plots 153

D-15 to D-20 Case 2 GM Performance Plots 167

D-21 to D-32 Case 3 GM Performance Plots 173

D-33 to D-36 Case 4 GM Performance Plots 185

D-37 to D-46 Case 5 GM Performance Plots 189

D-47 to D-62 Case 6 GM Performance Plots 199

D-63 to D-68 Case 7 GM Performance Plots 215

D-69 to D-76 Case 8 GM Performance Plots 221

D-77 to D-92 Case 9 Gm Performance Plots 229

D-93 to D-104 Case 10 Gm Performance Plots 245

D-105 to D-114 Case 11 GM Performance Plots 257

E-1 to E-16 Case 12 CTR Performance Plots . . . 270

E-17 to E-22 Case 13 CTR Performance Plots . . . 286

E-23 to E-38 Case 14 CTR Performance Plots . . . 292

E-39 to E-44 Case 15 CTR Performance Plots . . . 308

E-45 to E-52 Case 16 CTR Performance Plots . . . 314

E-53 to E-68 Case 17 CTR Performance Plots . . . 322

E-69 to E-72 Case 18 CTR Performance Plots . . . 338

E-73 to E-83 Case 19 CTR Performance Plots . . . 342

E-84 to E-89 Case 20 CTR Performance Plots . . . 353

E-90 to E-93 Case 21 CTR Performance Plots . . . 359

E-94 to E-97 Case 22 CTR Performance Plots . . . 361

E-98 to E-119 Case 23 CTR Performance Plots . . . 367

E-120 to E-135 Case 24 CTR Performance Plots . . . 389

Figures Pages

F-1 to F-8 BM Performance Plots - Trajectory 3 . . 406

F-9 to F-16 BM Performance Plots - 3 Spot Image . . 414

F-17 to F-24 BM Performance Plots - 3 Spot Image . . 422

F-25 to F-32 BM Performance Plots - 2G Pull-up . . . 430

F-33 to F-40 BM Performance Plots - 5G Turn 438

List of Tables

Table Page

3.1 Non-zero Correlation Coefficients 71

5.1 Target Trajectories Simulated 100

6.1 Time Averaged Figures of Merit-Dynamics Position Estimates 110

6.2 Time Averaged Figures of Merit-Centroid Position Estimates 111

6.3 Time Averaged Error Statistics when Tracking a Single Hot-Spot Image 116

6.4 Filter Performances with a Multiple Hot-spot Image 119

6.5 Performance vs Constant G Maneuvers 125

6.6 Performance vs Constant Roll-Rate Maneuvers . 128

D.1 Parameters Pertaining to Cases with GM Filter 151

E.1 Parameters Pertaining to Cases with CTF Filter 268

List of Symbols

Symbol	
$\vec{A}$	general inertial vector
$\vec{a}_I$	inertial acceleration
$\underline{a}$	image plane acceleration vector
A,B	atmosphere break frequencies
AR	aspect ratio
$\underline{A}$	direction cosine matrix
A_p	area of a single pixel
$\alpha(t)$	azimuth
$\beta(t)$	elevation
$\underline{B}$	control input matrix
$\underline{C}$, $\underline{C}_T$	selection matricies
δ_v, δ_{pv}	displacement of ellipsoid center from center of gravity in target frame (meters)
δ_α, δ_β	projection of above displacement onto $\alpha\ \beta$ plane
$\Delta\alpha$, $\Delta\beta$	corresponding angular displacements
$d_{k,l}$	distance (in pixels) between K-th and l-th pixels
$\underline{e}$	error between truth model system output and filter system output
$\vec{\underline{e}}$	unit vector
$\bar{E}$	mean error (ensemble averge)
$E[\cdot]$	expected value
f_x, f_y	spatial frequencies

List of Symbols

Symbol	
$\underline{f}[\quad]$	nonlinear function of filter states
$\underline{F}[\quad]$	linearized function of the filter states
$\underline{F}$	system plant matrix
$F\{\cdot\}$	Fourier Transform operation
$F^{-1}\{\cdot\}$	Inverse Fourier Transform operation
γ	angular displacement of craft in target frame
$\underline{G}$	system noise input matrix
$\tilde{g}$	complex-valued function in space domain
$\tilde{G}$	complex-valued function in spatial frequency domain
$h[\quad]$	nonlinear intensity function
$H[\quad]$	linearized intensity function
$\underline{H}$	system output matrix
$\vec{i},\vec{j},\vec{k}$	unit vectors along inertial axes
$\underline{I}$	identity matrix
Imax	maximum intensity
$\underline{K}$	Kalman filter gain
NxN	size of data array to be processed
Φ	system state transition matirx
θ	orientation angle in image plane
$\underline{P}$	state covariance matrix or dispersion matrix of target image
$\underline{Q}_A$	noise covariance kernel descriptor for truth model atmospherics
$\underline{Q}_{FD}$	dynamics driving noise matrix

Symbol	
$r_{k,l}$	correlation coefficient between k-th and l-th pixels
$\underline{r}$	residual vector
r_h	horizontal range
ρ	range
$\underline{R}$	measurement noise covariance matrix
σ	standard deviation of a process or dispersion of an intensity distribution
σ^2	variance of a process
t,τ	time
Δt	sample time
$\underline{u}$	deterministic control input
$\underline{v}$	measurement noise vector - zero mean, white, Gaussian noise
$\underline{v}'$	unit variance, zero mean, white, Gaussian noise vector
$\vec{v}$	inertial velocity vector
ω	constant turn-rate or roll-rate
$\underline{w}$	white Gaussian noise vector
x,y	space variables
(x,y)	FLIR frame coordinates
(x',y')	FLIR frame aligned with image velocity vector
$\underline{x}$	general state vector
$\hat{\underline{x}}$	estimated state vector
$\Delta\underline{x}$	state vector update
$\vec{X}_I,\vec{Y}_I,\vec{Z}_I$	inertial axes
$\hat{y}(t)$	current averaged data frame
$y(t)$	current data frame

Symbols

$\hat{y}(t-1)$	previous averaged data frame
α	smoothing constant in averaging process
$\underline{y}$	system output vector
$\underline{z}$	measurement vector
ζ	angle between inertial velocity vector and the plane that is perpendicular to the line of sight
$\perp$	perpendicular
$\sqrt[c]{\;}$	Cholesky square root
LOS	line of sight
RMS	Root Mean Squared

Subscripts

A,a	atmospheric jitter
α	aximuth direction
β	elevation direction
CEN	centroid
d	discrete form
D	target dynamics
F	filter
i	i-th time frame
I	inertial reference frame
kl	kl-th pixel
m	m-th hot-spot or ellipsoid
o	initial value
pl	planar
pm	intensity peak of m-th hot-spot

Symbol

Subscripts

pv	a direction perpendicular to the target velocity
ppv	a direction mutually perpendicular to the target velocity and to pv
r	direction along the line of sight to the target
T	truth model
v	direction of the target velocity vector

Superscripts

$\cdot$	time derivative
$\hat{}$	estimate
+	after measurement update
-	before measurement update
I	coordinatized in the inertial frame
$\rightarrow$	vector in inertial space
$\sim$	complex variable

Abstract

Two extended Kalman filter algorithms that estimate target position, velocity, and acceleration, as well as atmospheric jitter are developed for use within a laser weapon system. Digital signal processing techniques are employed on data obtained from a forward looking infrared (FLIR) sensor in order to identify the underlying shape of "multiple hot-spot" targets. No a priori information is assumed about such images. The identified shape is used in the measurement model portion of the extended Kalman filters in order to estimate target position offsets from the center of the sensor field of view. The two dynamics models incorporated within the filters are 1) a first order Gauss-Markov acceleration model and 2) a constant turn-rate acceleration model. Performance of these two filters are compared in tracking scenarios involving constant G and constant roll-rate maneuvers. Extensive consideration is given to simulating realistic multiple hot-spot images on the FLIR image plane. Performance of a previously developed adaptive filter is shown to be seriously degraded when faced with multiple hot-spot images since it assumes a priori information about the targt image. All evaluations are conducted using Monte Carlo techniques.

I. Introduction

Since their initial development, lasers have been used for many purposes throughout the medical, industrial, scientific and military fields. Such applications have varied widely from the use of low power lasers for precision surgery to the use of high energy lasers for their destructive potential. In particular, the high energy laser (HEL) has become very attractive to the United States Air Force for use as a key component of various weapon systems. An example of one such system is a ground-based, anti-aircraft/anti-missile weapon system. The advantages of a HEL system over conventional systems would include the ability to be used repeatedly and to deliver large amounts of energy over long distances essentially instantaneously.

Although the technology to produce and transmit lethal amounts of energy does exist, various other problems are still present. A major obstacle is the precision pointing of the laser and accurate tracking of the target. It is not sufficient simply to sweep the laser beam across the target; instead, it must be held at one location for some finite period of time. Factors that work against this are the motion of the target itself and the effect of the atmosphere on the propagation of the beam. Thus the realization of a HEL weapon system is dependent upon the development of a tracker capable of performing effectively against a variety of targets and environments.

1.1 Background

The basic components of a HEL weapon system are the laser itself, the sensor that supplies measurements and to which the laser is servoed (or shares an aperature with), and a control subsystem. One commonly used sensor, the one that will be referred to throughout this research, is the Forward Looking Infra-Red sensor (FLIR). The control subsystem consists of a tracker and gimbal controllers. The tracker utilizes sensor measurements to generate an estimate of the target position offset from the center of the field of view of the sensor, which is provided to the gimbal controllers to re-position the FLIR/laser. Traditionally, correlation algorithms alone have been used to provide the necessary position offset information to the gimbal controllers. This information was generated by comparing "templates" of predetermined or real-time-obtained target images with the most current set of measurements acquired from sensors. However, while being robust in the sense that it can operate successfully despite image shapes and environments that may vary widely, this algorithm has several major disadvantages: it is susceptible to noise, it is not sensitive to knowledge of target dynamics, and it does not account for the atmospheric distortion that radiated beams experience.

In order to overcome these deficiencies, a considerable amount of effort has been expended in recent years, at the

Air Force Institute of Technology, to develop an effective Extended Kalman Filter (EKF) algorithm for precision pointing and tracking (Ref: 3,5,12,16,17,19). In an initial feasibility study (Ref: 12), a simple EKF algorithm was designed to track a distant (point source) target exhibiting rather benign dynamics. Target estimates were based upon FLIR measurement data that was assumed to be corrupted by temporally and spatially uncorrelated noises. In this study, Mercier took into account the fact that apparent target location (from FLIR measurements) is actually the sum of the effects of true target dynamics and atmospheric distortion. Under the assumed dynamics conditions, the resulting four-state EKF (two target position states and two atmospheric jitter states describing motion on the two-dimensional FLIR image plane) consistently outperformed correlation trackers by an order of magnitude in regard to RMS tracking errors. It should be noted, however, that an assumption was made that the target image could be well described analytically as a two-dimensional Gaussian intensity profile with circular equal-intensity contours. This assumption is reasonable only as long as the target is very distant.

Subsequently, robustness studies (Ref: 5, pp 45-46) indicated that the four-state filter was inadequate when actual target dynamics and target range were much different than that assumed in its derivation. Thus, adding target velocity and acceleration states to the original four, Captains Harnly and Jensen developed an eight-state adaptive EKF that enabled tracking of more maneuverable targets at closer ranges. While still assuming the target image could be described analytically, it was portrayed as a bivariate gaussian intensity distribution with elliptical equal-intensity contours. However, provisions were made to estimate the size and shape parameters of the elliptical contours adaptively. In addition, Harnly and Jensen more accurately modelled spatial and temporal correlations of backgound noise to enable better separation of image plane position offsets due to atmospheric jitter from those due to target dynamics. Overall, the filter was found to be very robust, even when tracking maneuverable targets at close ranges, as long as the target image could be well described analytically as a bivariate Gussian intensity distribution.

This filter was then taken and used as the basis for comparing different dynamics models as well as a multiple model adaptive algorithm (Ref: 3). The dynamics models considered were the original Brownian motion (BM) acceleration model and a constant turn-rate (CTR) acceleration model. The multiple model filter consisted of a bank of BM filters tuned to handle different degrees of target dynamics. Analysis indicated that the CTR dynamics model outperformed the other filters by reducing mean bias errors. A major problem with the multiple model algorithm was its inability to select one of its bank of filters over the others because of the similarity of the residuals from all filters.

Along another line, work has been accomplished to illustrate the feasibility of a multiple hot-spot tracker that does not assume a priori knowledge of the shape of the target image (Ref: 16,17). Digital signal processing techniques were used on the FLIR data to determine the underlying shape of the target image. This reference image, along with its derivatives, are required in the measurement update portion of the extended Kalman filter. In an initial effort (Ref: 17), the reference image derivatives were approximated using the Forward-Backward Difference Method. However, resulting filter divergence problems were not able to be overcome. Since then, a study performed by 1Lt Steven K. Rogers successfully demonstrated the potential of the multiple hot-spot tracker (Ref: 16). In this case, the derivative property of the Fourier Transform was implemented to determine the reference image derivatives. However, since concept feasibility of a multiple hot-spot tracker was the goal of Rogers' research, a four-state filter was used that only estimated target position and jitter variables, as in the research by Mercier. In addition, the multiple hot-spot target was modelled simply as a static image that would be realistic for the trajectory of a target flying a radial path about the tracker; not particularly realistic, but able to demonstrate concept feasibility.

1.2 Problem

This leads to the specific objectives of this research. The main goal was to implement the digital signal processing techniques inherent in the multiple hot-spot tracking algorithm (Ref: 16), which assumes no a priori knowledge of the target image, as part of an eight-state extended Kalman filter that estimates target velocity and acceleration as well as position and atmospheric jitter. To be exact, two dynamics models are considered. They portray the FLIR image plane acceleration as either (1) a first order Gauss-Markov process or (2) a constant turn-rate process. The former describes the time derivative of acceleration via the relation

$$\dot{\underline{a}}(t) = -\frac{1}{\tau_D}\,\underline{a}(t) + \underline{w}(t) \tag{1-1}$$

where τ_D represents the correlation time and $\underline{w}(t)$ is a white Gaussian noise vector. In contrast, the constant turn-rate model is defined by replacing Eq (1-1) with

$$\dot{\underline{a}}(t) = -\omega^2\underline{v}(t) + \underline{w}(t) \tag{1-2}$$

where the parameter ω is the target image turn-rate and can be expressed as

$$\omega = \frac{|\underline{v}(t) \times \underline{a}(t)|}{|\underline{v}(t)|^2} \tag{1-3}$$

The details of these models can be found in Chapter IV; however, it should be noted that the purpose of comparing the two is to see whether the improvement expected in the performance of the constant turn-rate model is worth the additional on-line computation required due to the nonlinear dynamics of Eqs (1-2) and (1-3) instead of the linear time-invariant dynamics associated with Eq (1-1).

In order to test the effect of the digital signal processing techniques on tracker performance thoroughly, another objective of this research was to simulate a realistic changing multiple hot-spot image. Extensive consideration was given to the dynamic sizes, shapes, and spatial inter-relationships of the hot-spots on the image plane. Also of importance was simulating the disappearance of portions of the target image due to obscurations of parts of the target from the sensor by other portions of the target itself.

Whatever shape the target image displays, it is critical that the extended Kalman filter obtain an accurate reference image, $\underline{h}[\hat{\underline{x}}(t_i), t_i]$, as well as the derivatives of that image, $\underline{H}[\hat{\underline{x}}(t_i), t_i] = \partial \underline{h}[\hat{\underline{x}}(t_i), t_i]/\partial \underline{x}$, with respect to the states. After propagating its state estimates forward from one sample time to the next, the filter processes the reference image, along with the measurement vector, in order to produce an updated state estimate using the equation

$$\hat{\underline{x}}(t_i^+) = \hat{\underline{x}}(t_i^-) + \underline{K}(t_i)[\underline{z}(t_i) - \underline{h}(\hat{\underline{x}}(t_i^-),t_i)] \qquad (1\text{-}4)$$

where

$\hat{\underline{x}}(t_i^+)$ = estimated state vector after incorporation of the measurement of time t_i

$\hat{\underline{x}}(t_i^-)$ = estimated state vector after propagation from the previous measurement update to time t_i

$\underline{K}(t_i)$ = Kalman filter gain

$\underline{z}(t_i)$ = 64-dimensional measurement vector of the average intensities of the pixels within the 8x8 field of view of the FLIR sensor

$\underline{h}(\hat{\underline{x}}(t_i^-),t_i)$ = nonlinear intensity function of the target image at time t_i as a function of the state estimate.

The block diagram shown in Figure 1 illustrates the data processing algorithm used in this research. Essentially, there are two parallel data processing paths in regard to the FLIR intensity measurements. In the lower path, the 8x8 array of intensity measurements are rearranged by rows into a single 64x1 measurement vector, $\underline{z}(t_i)$. Then the eight-state extended Kalman filter uses the nonlinear and linearized intensity functions provided by the upper path in order to produce an updated state estimate, $\hat{\underline{x}}(t_i^+)$. Then using its internal dynamics model, the filter propagates the state estimate forward in time to produce $\hat{\underline{x}}(t_{i+1}^-)$. The position components of this vector resulting from target dynamics are then fed to a controller in order

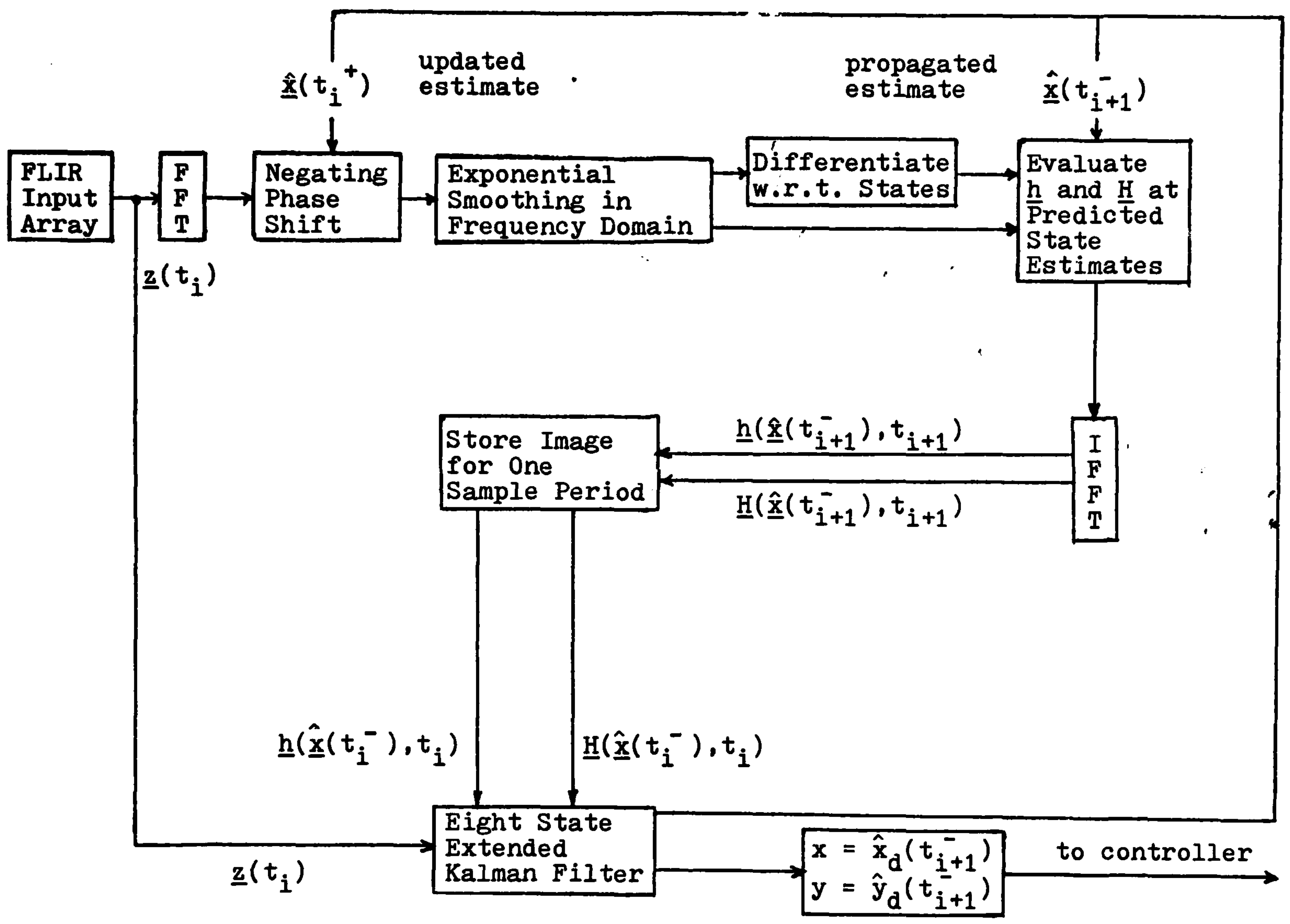

FLIR Input Array
FFT
$\underline{z}(t_i)$
Negating Phase Shift
$\hat{\underline{x}}(t_i^+)$
updated estimate
Exponential Smoothing in Frequency Domain
Differentiate w.r.t. States
propagated estimate
$\hat{\underline{x}}(t_{i+1}^-)$
Evaluate $\underline{h}$ and $\underline{H}$ at Predicted State Estimates
IFFT
$\underline{h}(\hat{\underline{x}}(t_{i+1}^-),t_{i+1})$
$\underline{H}(\hat{\underline{x}}(t_{i+1}^-),t_{i+1})$
Store Image for One Sample Period
$\underline{h}(\hat{\underline{x}}(t_i^-),t_i)$
$\underline{H}(\hat{\underline{x}}(t_i^-),t_i)$
Eight State Extended Kalman Filter
$\underline{z}(t_i)$
$x = \hat{x}_d(t_{i+1}^-)$
$y = \hat{y}_d(t_{i+1}^-)$
to controller

Figure 1. Data Processing Algorithm

to position the FLIR so that its center will be located at the filter-predicted location of the target at the next sample time. The updated and propagated state estimates are also provided to the upper data processing path for use in determining the centered nonlinear and linearized intensity functions.

Specifically, the first step in the upper path is to take the two-dimensional Fourier Transform of the FLIR measurement data. However, now a 24x24 array is processed (as opposed to an 8x8 array) in order to reduce edge effects, aliasing, and leakage conditions involved when transforming finite sequences (Ref: 17, p 18) . Also, rather than simply pad the data with zeros, as is often done in engineering practices, actual data is used. It would be appropriate to pad with zeros if the image intensity at the edges of the field of view were essentially zero. However, since this is often not the case, padding with zeros introduces artificial edge effects. The luxury of being able to pad with data is available in this application since the field of view occupies only a small portion of a much larger measurement array.

Since the state estimates that the Kalman filter feeds to the controller are only intended to zero out the effects of predicted target dynamics, the observed target image is expected to be offset from the center of the field of view by atmospheric jitter plus an amount due to the imperfect propagation of the dynamic states. In order to acquire the effect of a centered image in the original spatial domain, the Fourier Transform of the measurement data is multiplied by a negating phase shift in the frequency domain. The appropriate phase shift is determined by employing the Shift Theorem along with the filter's updated state estimate as determined by Eq (1-4). This produces the negating shift which is the complex conjugate of the linear phase shift that corresponds to the image offset expected in the space domain.

Once the image is centered, inter-frame smoothing of sequential transformed images is performed in order to reduce the effects of noise corruption (i.e., to make the underlying target image more obvious based upon the idea that the target intensity pattern will not change as much, from one sample time to the next, as will the noise). This is accomplished through the use of an exponential smoothing algorithm. The centered and smoothed data is then differentiated with respect to the Kalman filter position states by employing the Derivative Property of the Fourier Transform. This produces the frequency domain representation of the linearized intensity function, $\underline{H}$.

Ultimately, the nonlinear and linearized intensity functions are to be used in updating the state estimate after the next measurement. Therefore, they must first be evaluated in terms of the propagated state estimate, $\hat{\underline{x}}(t_{i+1}^-)$. Since it is assumed that the FLIR will be centered at the predicted position due to target dynamics, the Shift

Theorem is applied to each of the intensity functions to produce a shift equivalent to only the atmospheric states. Finally, after evaluating the inverse Fourier Transform of the results, $\underline{h}[\hat{\underline{x}}(t_{i+1}^-), t_{i+1}]$ and $\underline{H}[\hat{\underline{x}}(t_{i+1}^-), t_{i+1}]$ are ready for use by the extended Kalman filter in processing the next measurement frame.

Assumptions

FLIR. Although data is generated in the computer simulation in this research, it is assumed that real data would be available as the outputs of a FLIR sensor. The FLIR produces a frame of data every thirtieth of a second (30 Hz frame rate). Each frame consists of about 500x400 pixels (picture elements) of information. However, only 8x8 arrays ("tracking windows") for measurement updating and 24x24 arrays for data processing are manipulated in this research in order to limit computational and storage requirements (Ref: 5, p 4). Changing the field of view, i.e., changing the size of the tracking window, is a realistic possibility for handling harshly maneuvering targets; however, this aspect is not investigated in this research.

Background Noise. The background noise that corrupts the FLIR measurements is modelled as a spatially correlated Gaussian process. Although different physical backgrounds will result in different spatial correlations, real data analysis (Ref: 5) has indicated that a reasonable model of the spatial correlation is a decaying exponential that is non-zero for pixel distances that include the first and second nearest neighbors (i.e., the correlation function drops an order of magnitude exponentially in a distance equal to two pixels).

Ground Based. Since the FLIR is part of a ground based system, mirror and base vibrations are assumed negligible and thus make no contribution to pointing errors.

Closed Loop. Since the FLIR, the laser, and the controlling subsystem form a closed loop system, it is assumed that the pointing system is perfect. That is, the FLIR/laser can be pointed to whatever spot the tracker commands within the time available between measurements.

Overview

Immediately following this section is Chapter II which describes the data processing techniques employed in Figure 1 in more detail. This is followed by the truth model development in Chapter III, which presents the mathematical models used to represent the real world so that a performance analysis of the filters can be conducted. Chapter IV presents the two different filter models studied in this research: the first order Gauss-Markov and the constant turn-rate models. Also in Chapter IV is a discussion of estimation of the dynamic driving noise in the system. This is followed, in Chapter V, by a description of the performance analysis methodology used to test the filters developed in Chapter IV against the truth model described in Chapter III. Chapter VI presents the results of the performance analysis, and this is followed by conclusions and recommendations in Chapter VII.

II. Derivation of Intensity Functions

2.1 Introduction

In order to track a wide variety of targets which may exhibit unknown and/or changing sizes and shapes on the FLIR image plane, the EKF must be able to generate an accurate intensity profile of the target image, $\underline{h}[\hat{\underline{x}}(t_i^-), t_i]$, as well as the derivative of that profile, $\underline{H}[\hat{\underline{x}}(t_i^-), t_i]$, with respect to the states. It is the information within these two functions, evaluated at the current state estimate, that the EKF processes along with noise corrupted FLIR measurements in order to compute updated state estimates, as in Eq (1-4). Specifically, the goal is to recognize the true intensity function of the target image using past noise-corrupted data frames.

In theory, it is possible to represent all the information in a two-dimensional intensity function by a set of eigenfunctions and corresponding eigenvalues. However, an exact representation could well require an infinite number of such functions and values. Thus, a well suitable transformation must be determined that is not overly cumbersome yet yields an accurate representation of patterns within the intensity data.

One possible method is known as the Karhunen-Loeve Transformation. This transformation generates a new coordinate space with totally uncorrelated image components based on the properties of the spatial autocorrelation kernel (Ref: 17, p 9). For an NxN input matrix, the eigenvalues correspond to actual variance statistics projected onto N^2 orthogonal eigenfunctions. Thus, this transformation has the disadvantage of requiring the manipulation of an $N^2 x N^2$ correlation matrix whose exact calculation is often very difficult to perform.

Alternatively, if the assumptions of spatial stationarity and a space domain large in extent are made, the Karhunen-Loëve Transformation provides motivation for the use of the Fourier Transform (Ref: 17, p 12). Although the Fourier Transform does not produce perfect decorrelation of new data components, it is desirable computationally and also because of the property of separability that enables the two-dimensional transform to be obtained through simpler one-dimensional operations.

2.2 Two-Dimensinal Fourier Transform

As in the case of the one-dimensional transform, the two-dimensional Fourier Transform utilizes complex exponentials as eigenfunctions; however, now two spatial frequencies (f_x and f_y) are generated which are related to the spatial coordinates, x and y respectively. Thus, the Fourier Transform of a complex-valued function of independent variables, $\tilde{g}(x,y)$, can be regarded as a decomposition of that function into a linear combination of elementary functions of the form $\exp[j2\pi(xf_x + yf_y)]$ (Ref: 4, p 8). The transform will be represented as $F(\tilde{g})$ and defined by

$$F(\tilde{g}) = \tilde{G}(f_x , f_y)$$

$$= \int_{-\infty}^{\infty} \tilde{g}(x,y)\exp[-j2\pi(xf_x + yf_y)]dxdy \qquad (2\text{-}1)$$

where

$\tilde{G}(f_x,f_y)$ = Frequency Spectrum

$\tilde{g}(x,y)$ = Function in Spatial Domain

f_x,f_y = Spatial Frequencies

x,y = Spatial Variables

The transform itself, $\tilde{G}(f_x,f_y)$, is also a complex-valued function, but of the independent spatial frequencies, f_x and f_y.

In order for any transform to be useful, its inverse must also exist. The inverse Fourier transform of a function $\tilde{G}(f_x,f_y)$ is defined as

$$F^{-1}(\tilde{G}) = \tilde{g}(x,y)$$

$$= \int_{-\infty}^{\infty} \tilde{G}(f_x,f_y) \exp[j2\pi(xf_x + yf_y)]df_x df_y \qquad (2\text{-}22)$$

where the terms are as defined above. The equations (2-1) and (2-2) are defining relations for the continuous Fourier Transform; however, in this research data is available as a finite number of discrete values. Thus it is necessary to process the data using the two-dimensional finite Discrete Fourier Transform (DFT). The development of the DFT can be found in Digital Signal Processing by Oppenheim and Schafer (Ref: 15) with the result as presented here. Thus for an NxN set of discretized intensity values,

$$\tilde{G}(f_x, f_y) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \tilde{g}(x,y) \exp[-j(2\pi/N)(x f_x + y f_y)] \qquad (2\text{-}3)$$

with the corresponding inverse Discrete Fourier Transform given by

$$\tilde{g}(x,y) = (1/N^2) \sum_{f_x=0}^{N-1} \sum_{f_y=0}^{N-1} \tilde{G}(f_x, f_y) \exp[j(2\pi/N)(x f_x + y f_y)] \qquad (2\text{-}4)$$

where $\tilde{g}(x,y)$ and $\tilde{G}(f_x,f_y)$ now represent discrete NxN arrays. As in the continuous case, the discrete transform is separable and can be obtained through simpler one-dimensional operations.

For implementation in the computer software, the DFT is replaced by a more efficient version known as the Fast Fourier Transform (FFT). The FFT and its inverse (IFFT) are performed by subroutine FOURT (see Appendix G). FOURT utilizes what is known as the Cooley-Tukey method of calculating the FFT of multi-dimensional arrays. Further information on the routine can be found in the commented software itself; or for more detail, see (Ref: 2, pp 76-9) or Ref: 16, pp 131-45).

2.3 Exponential Smoothing

In order to update the state equations, the extended Kalman filter must generate a representation of the target's intensity pattern on the FLIR image plane. Several things work against the filter being able to generate an accurate profile. They include the presence of noise corruption within the raw measurements and the fact that a realistic target image will display continuous size and shape changes. However, at reasonable measurement rates, even a dynamic image will not vary as fast, from sample period to sample period, as does the corruptive noise. Thus, interframe smoothing can be utilized to reduce the effects of unwanted noise.

One method that is traditionally used to combat noise effects is the moving average technique. This technique is simply an average of the K most recent frames of data where the value K is chosen to reflect how long past frames will be considered in the current average. However, KN^2 storage locations are required in order to average K NxN data arrays. A less costly technique is the exponential smoothing algorithm which is represented by the following equation (Ref: 1):

$$\hat{y}(t) = \alpha y(t) + (1-\alpha)\hat{y}(t-1) \qquad (2\text{-}5)$$

where

$\hat{y}(t)$ = the most recent averaged frame
$y(t)$ = the current data frame
$\hat{y}(t-1)$ = the previos averaged data frame
α = the smoothing constant such that $0 \leq \alpha \leq 1$

As indicated, the smoothing constant, α, can vary anywhere from 0 to 1, inclusive, depending on the weight that is to be put on the current data frame. For static and slowly varying images, it is desirable to use a steady state smoothing constant that is of very small magnitude in order to obtain maximum benefit from the smoothing process. However, such a value would have to be somewhat larger for more dynamically changing images, since there is a need to weight the current data more. A tradeoff analysis must be conducted in order to determine a smoothing constant suitable for all degrees of image variation to be encountered.

For implementation, it is appropriate that the smoothing constant change during initial time frames. Specifically, $\alpha = 1/K$ for $K = 1,2,3\ldots$ until the desired steady state value is reached. In addition, the smoothing process itself can be performed in either the spatial or frequency domains. Equivalent results have been achieved using either approach (Ref: 16, p 99). However, since smoothing in the frequency domain required fewer transformations, and thus fewer numerical errors, it was chosen for use in this research.

In summary, it should be noted that the exponential smoothing technique uses only $2N^2$ memory locations to generate the most recent averaged data frame, $\hat{y}(t)$, compared to the KN^2 locations required by the moving average technique. In addition, $\hat{y}(t)$ contains some information from all previous data frames although, due to repeated application of the weighting factor, initial frames have much less effect on the averaged data than the current frame.

2.4 Shifting Property

In order to determine the underlying intensity profile of the target image from noise corrupted data, exponential smoothing is used as discussed above in order to determine which portion of the observed profile is common to all data frames. However, before this can be accomplished, the pattern must first be centered since the pattern will experience different shifts from frame to frame. It is towards this end that the shifting property of the Fourier Transform is used, along with the filter's estimate of the target location on the image plane.

The shift theorem states that the translation of a function in the space domain introduces a linear phase shift in the spatial frequency domain (Ref: 4, p 9). The Fourier

Transform views a finite length sequence of length N as a single period of a corresponding periodic sequence of period N. Therefore, a shift on a finite length discrete one-dimensional sequence can be thought of as a rotation of samples within the basic interval of the sequence. That is, as samples are shifted out one side of the interval, identical samples enter the other side (Ref: 15, p 102). Such a shift is often termed a cylindrical shift since if the samples were to lie on the circumference of a cylinder a linear shift would correspond to a rotation of the cylinder.

These ideas can be extended to the case of a finite area two-dimensional sequence. In relation to Eq (2-1) a translation of the complex valued function, $\tilde{g}(x,y)$, in the space domain results in the following Fourier spectrum in the spatial frequency domain

$$F(\tilde{g}(x-a,y-b) = \tilde{G}(f_x,f_y)\exp[-j2\ (af_x + bf_y)] \qquad (2\text{-}6)$$

where

$$F(\tilde{g}(x,y)) = \tilde{G}(f_x,f_y)$$

a = shift of the spatial function in the x direction

b = shift of the spatial function in the y direction

Equation (2-6) indicates that the only difference between the original spectrum and the spectrum corresponding to the translated function is the addition of a linear phase shift that is proportional to the magnitude of the spatial translations in the x and y directions. As a result, if the magnitude of the spatial shift is known, a negating phase shift can be applied to the original transform in order to obtain the transform of an equivalent but centered image. Specifically, the steps to be performed to obtain the centered intensity profile are as follows:

- obtain the transform of the offset intensity profile (raw meaurements)
- multiply the transform by the complex conjugate of the phase shift introduced in the frequency domain; this negating phase shift is formed using the extended Kalman filter's estimate of the image location
- implement smoothing algorithm if smoothing is to be done in the frequency domain
- take the inverse transform to obtain the centered intensity profile.

The equation describing these steps is

$$\tilde{g}(x,y) = \{F^{-1}(F[\tilde{g}(x-a,y-b)]\exp[+j2\pi(af_x + bf_y)]\} \tag{2-7}$$

where a,b now represent the filter's estimates of the image offsets in the x and y directions respectively, and, the frequency domain smoothing is accomplished prior to taking the inverse transform.

2.5 Derivative Property

One of the reasons that data processing in the frequency domain is so desirable is because of the derivative property of the Fourier Transform. As indicated in the overview, the extended Kalman filter requires an accurate representation of not only the nonlinear intensity function, $\underline{h}[x(t_i), t_i]$, but of the linearized intensity function, $\underline{H}[\underline{x}(t_i), t_i] = \partial \underline{h}[\underline{x}(t_i), t_i]/\partial \underline{x}$, as well. Using the Fourier Transform, differentiation in the spatial domain reduces to a simple multiplication in the spatial frequency domain. Attempts have been made to implement a simpler method known as the Forward-Backward Difference Method (Ref: 17). However, filter divergence problems indicated the need for a better technique. In contrast, use of the Fourier Transform derivative property is more accurate and has been effectively implemented (Ref: 16) because it utilizes the entire data array to compute the derivatives as opposed to just the values in the preceding and subsequent pixels. Analytically, the process can be described by

$$F\left[\frac{\partial \underline{h}(x,y)}{\partial x}\right] = j2\pi f_x \; F[\underline{h}(x,y)] \tag{2-8a}$$

$$F\left[\frac{\partial h(x,y)}{\partial y}\right] = j2\pi f_y \; F[\underline{h}(x,y)] \tag{2-8b}$$

where $\underline{h}(x,y)$ represents the centered and smoothed intensity profile. Implementation of the derivative property is accompished in Subroutine DERIV (see Appendix G) where each data point in the transform domain is multiplied by (j2 times the spatial frequency associated with both that location as well as the direction of the desired derivative).

2.6 Summary

This section has presented the signal processing techniques behind the recognition and generation of the nonlinear, $\underline{h}[\hat{\underline{x}}(t_i^-), t_i]$, and linearized, $\underline{H}[\hat{\underline{x}}(t_i^-), t_i]$, intensity functions. These functions are required by the extended Kalman filter in the measurement update portion of state estimation (to be discussed in Chapter IV). To acquire these functions, the filter must process raw measurement data. This data contains the intensity profile of the target which is corrupted by noise and is shifted from the center of the field of view by atmospheric jitter as well as by filter tracking errors. In order to obtain the desired intensity functions, the raw data is first transformed to the spatial frequency domain via the Fourier Transform. Since the data is available only as discrete values, the Discrete Fourier Transform, or actually a more efficient version known as the Fast Fourier Transform, is used in the actual implementation. The transformation itself can be regarded as the decomposition of the input array into a linear combination of complex exponential eigenfunctions which indicate the spatial frequency content of the signal.

Once the data is expressed in the spatial frequency domain, smoothing is performed in order to reduce the effects of noise. An exponential smoothing algorithm was chosen for this inter-frame smoothing process since it requires much less memory compared to other techniques. The algorithm is flexible in the sense that the steady-state smoothing constant can be chosen small to take full advantage of the smoothing process when the target image intensity distribution is relatively static compared to the rate at which measurements are made. In contrast, the smoothing constant should be somewhat larger for more dynamic image variations.

Since the intensity pattern that is common to successive data frames will have experienced different spatial offsets, it must be centered each frame prior to smoothing. This is accomplished via the Shifting Theorem of the Fourier Transform. Using this theorem, the offset in the intensity profile can be corrected by an amount equal to the filter's estimate of the atmospheric jitter.

Once the image has been centered and smoothed, the spatial derivatives of the profiles can be generated by implementing the Derivative Property of the Fourier Transform in the frequency domain. These derivatives are required in order to determine the linearized intensity functions which, along with the centered nonlinear intensity function, are to be evaluated at the current state estimate for use in further processing by the filter. Since the high energy laser, and thus the center of the field of view, is to be located at the filter's best estimate of the target's location, the resulting intensity functions should be offset from the center by an amount equivalent to the estimated atmospheric jitter states. This is accomplished by again implementing the Shift Theorem prior to taking the inverse FFT. The final intensity functions are then used by the extended Kalman filter during the next measurement update.

III. Truth Model

3.1 Introduction

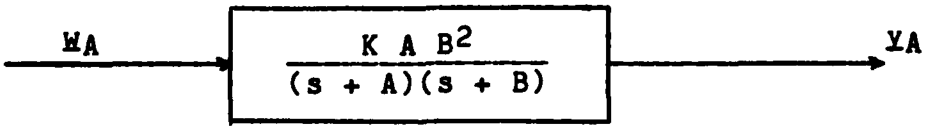
In most large scale design work, it is not feasible to build and test a product, in the real world, immediately after completion of the initial design. However, some means of evaluation must be used to determine if a design is suitable for eventual implementation. Thus in this research, a performance (sensivity) analysis is conducted (see Chapter V) to determine how well the filter designs might be expected to perform in the real world. Such an anlysis requires the replacement of the real world with a "truth model" which represents the best, most complete mathematical representation of the real world available to the designer. This chapter presents the various components of the truth model used to represent environmental characteristics such as atmospheric jitter effects, target dynamics, as well as FLIR and background noises. Extensive consideration is also given to simulating the measurement vector, $\underline{z}(t_i)$, of a realistic target image.

The initial two sections (3.2 and 3.3) describe the image plane translation of the target image intensity function due to atmospheric jitter as well as deterministic target dynamics. The following section (3.4) then formulates the state space representation of the entire system and describes its propagation in time. Section 3.5

describes the pixel by pixel creation of the measurement data for a target image with M hot-spots. This is followed in Section 3.6, by a description of the size, shape, and spatial inter-relationship of the M hot-spots involved in a dynamic image. Finally, the spatial correlation model of the background noise is presented in Section 3.7.

3.2 Atmospheric Disturbance Model

As a result of atmospheric phase front distortion, the intensity distribution of radiated wavefronts undergoes translational position changes with time on the FLIR image plane. An accurate representation of these turbulent effects was developed by The Analytic Sciences Corporation (TASC) based on data supplied by the Air Force Weapons Laboratory (Ref: 6). The resulting power spectral density (PSD) representation can be approximated by the power spectral density that is characteristic of the output of a third order shaping filter driven by zero-mean, white, Gaussian noise, as shown in Figure 2.



A = 14.14 rad/sec
B = 659.5 rad/sec

Figure 2. Third Order Shaping Filter

$\underline{w}_A$ = Zero-mean, white, Gaussian noise

A,B = Break frequencies

K = System gain - adjusted to obtain desired RMS jitter

$\underline{y}_A$ = System output that has the desired PSD

Since the effects of atmospheric jitter are independent of direction on the FLIR image plane, the resulting planar motion can be modelled as the output of two such shaping filters (corresponding to motion in each of the x and the y directions on the FLIR plane). Thus, the effects of the atmosphere can be described by a stochastic differential equation of the form:

$$\dot{\underline{x}}_A(t) = \underline{F}_A(t)\underline{x}_A(t) + \underline{G}_A(t)\underline{w}_A(T) \qquad (3\text{-}1a)$$

$$\underline{y}_A(t) = \underline{H}_A(t)\underline{x}_A(t) \qquad (3\text{-}1b)$$

where:

$\underline{x}_A(t)$ = the atmospheric noise states (three for each direction; six total)

$\underline{F}_A(t)$ = the atmospheric plant matrix

$\underline{G}_A(t)$ = the atmospheric noise input matrix

$\underline{w}_A(t)$ = two dimensional vector of zero-mean, white, Gaussian noise inputs

such that:

$\underline{y}_A(t)$ = the desired output of the system (shift in FLIR coordinates)

$\underline{H}_A(t)$ = the system output matrix

$$E[\underline{w}_A(t)] = \underline{0}$$

and

$$E[\underline{w}_A(t)\underline{w}_A(t+\tau)] = \underline{Q}_A(t)\,\delta(\tau) \qquad (3\text{-}2)$$

$\underline{Q}_A$ in the above equation is the atmospheric noise covariance kernel descriptor and is often assumed constant for off-line tuning. The results can then be implemented in the actual filter as constants or as the initial values of a time varying descriptor. The latter is appropriate in two instances: during the acquisition phase and when adaptive on-line tuning is implemented.

Using the Jordan canonical form to represent the system description of atmospheric turbulence in a single direction (Ref: 12, pp 73-75), Equation (3-1) corresponding to the x direction becomes:

$$\dot{\underline{x}}_{Ax}(t) = \begin{bmatrix} -A & 0 & 0 \\ 0 & -B & 1 \\ 0 & 0 & -B \end{bmatrix} \begin{bmatrix} x_{Ax1}(t) \\ x_{Ax2}(t) \\ x_{Ax3}(t) \end{bmatrix} + \begin{bmatrix} G_1 \\ G_2 \\ G_3 \end{bmatrix} w_{Ax}(t) \qquad (3\text{-}3)$$

where A,B are the break frequencies and

$$G_1 = \frac{KAB^2}{(A-B)^2}$$

$$G_2 = -G_1 \qquad (3\text{-}4)$$

$$G_3 = (A-B)G_1$$

Thus both $\underline{E}_{Ax}$ and $\underline{G}_{Ax}$ are constant matrices. This yields an output equation of the form:

$$y_{Ax}(t) = [\ 1 \quad 1 \quad 0\ ]\ \underline{x}_{Ax}(t) \qquad (3\text{-}5)$$

The general solution of the six states of Eq (3-1a) between one sample time and the next, i.e., for all $t \epsilon [t_i, t_{i+1}]$ is:

$$\underline{x}_A(t) = \underline{\Phi}_A(t,t_i)\underline{x}_A(t_i) + \int_{t_i}^{t} \underline{\Phi}_A(t,\tau)\underline{G}_A(\tau)\underline{w}_A(\tau)d\tau \qquad (3\text{-}6)$$

where $\underline{\Phi}_A(t,t_i)$ is the state transition matrix for the atmospherics which must satisfy the matrix differential equation:

$$\dot{\underline{\Phi}}_A(t,t_i) = \underline{E}_A\underline{\Phi}_A(t,t_i) \qquad (3\text{-}7)$$

with the boundary condition: $\underline{\Phi}(t_i,t_i) = \underline{I}$. When the plant matrix is constant, the state transition matrix becomes a function of $\Delta t = t_{i+1} - t_i$ for a fixed sampling time. More specifically,

$$\underline{\Phi}_A(t_{i+1},t_i) = \underline{\Phi}_A(\Delta t) \tag{3-8}$$

Thus for the plant in Equation (3-3) the state transition matrix for distortion in the FLIR x direction is:

$$\underline{\Phi}_{Ax}(t_{i+1},t_i) = \begin{bmatrix} e^{-A\Delta t} & 0 & 0 \\ 0 & e^{-B\Delta t} & \Delta te^{-B\Delta t} \\ 0 & 0 & e^{B\Delta t} \end{bmatrix} \tag{3-9}$$

An identical result follows for distortion in the FLIR y direction. For digital computer implementation it is necessary to develop equivalent discrete-time representations for the continuous-time processes (Ref: 7). Toward this end, a discrete-time noise process can be defined with statistics equivalent to those of the stochastic integral in Eq (3-6), i.e.

$$\underline{w}_{Ad}(t_i) = \int_{t_i}^{t_{i+1}} \underline{\Phi}_A(t_{i+1},\tau)\underline{G}_A(\tau)\underline{w}_A(\tau)d\tau \tag{3-10}$$

such that $E[\underline{w}_{Ad}(t_i)] = \underline{0}$

and

$$E[\underline{w}'_{Ad}(t_i)\underline{w}'^{T}_{Ad}(t_j)]$$

$$= \int_{t_i}^{t_{i+1}} \underline{\Phi}_A(t_{i+1},\tau)\underline{G}_A(\tau)\underline{Q}_A(\tau)\underline{G}_A^T(\tau)\underline{\Phi}_A^T(t_{i+1},\tau)d\tau \triangleq \underline{Q}_{Ad}(t_i)$$

$$(\text{for } t_i = t_j)$$

$$= \underline{0} \quad (t_i \neq t_j) \qquad (3\text{-}11)$$

Therefore, a discrete-time equation equivalent to Eq (3-6) can be written:

$$\underline{x}_A(t_{i+1}) = \underline{\Phi}_A(t_{i+1},t_i)\underline{x}_A(t_i) + \sqrt[c]{\underline{Q}_{Ad}}\underline{w}_{Ad}(t_i) \qquad (3\text{-}12)$$

where $\sqrt[c]{\underline{Q}_{Ad}}$ is the lower triangular Cholesky square root of $\underline{Q}_{ad}$ (Ref: 7), and,

$$E[\underline{w}'_{Ad}(t_i)] = \underline{0} \qquad (3\text{-}13)$$

$$E\underline{w}'_{Ad}(t_i)\underline{w}'^{T}_{Ad}(t_i)] = \underline{I}\delta_{ij}$$

Note that the properties of $\sqrt[c]{\underline{Q}_{Ad}}\underline{w}'_{Ad}$ are identical to those of $\underline{w}_{Ad}$ in Eqs (3-10) and 3-11). The details of the exact integration of (3-11) are shown in Appendix A.

3.3 Target dynamics Model - Trajectory Generation

The truth model for the target dynamics is a continuous-time deterministic model used to describe a highly maneuverable aircraft or missile. It is desired that the target simulations be as realistic as possible yet remain computationally feasible for on-line operation.

Therefore, in this research, the performance of the filter will be tested against, basically, three types of maneuvers:

(1) straight line propagation
(2) constant roll-rate maneuvers
(3) constant G, constant speed turns

Initially, these maneuvers will be examined separately to note their effects on the intensity distributions of the multiple hot-spot target on the FLIR image plane and on tracking performance. Then, combinations of these maneuvers will be pieced together. A typical trajectory might involve a straight line path while undergoing a constant roll-rate followed by a constant G pull-up.

The assumed target geometry will be described in detail in a subsequent section (FLIR Measurements); however, it should be noted that the center of gravity of the vehicle (about which the rolls will take place) is assumed to coincide with the centroid of the FLIR image when projected onto the FLIR plane. Thus, a constant roll maneuver, while on a straight line trajectory, will not affect the dynamics of the centroid. This is not the case, however, when the craft is undergoing a rolling, G-pulling maneuver.

All maneuvers will be simulated by implementing a predetermined deterministic flight path in inertial space. The target's true location is then projected onto the FLIR image plane for comparison with filter estimates in order to evaluate filter's tracking performance. The vehicle's flight path can be generated through the use of control inputs to the truth model dynamics similar to the method used by Captains Harnly and Jensen (Ref: 5). One major difference is that, in this research, azimuth and elevation acceleration inputs are generated for use in propoagating the true position as well as determining the performance of the filter's acceleration estimate. Thus, a convenient means of simulating a prescribed position time history is through the use of the following vector dynamics differential equation:

$$\dot{\underline{x}}_D(t) = \underline{B}_D\underline{u}_D(t) = \begin{bmatrix} \dot{\alpha}(t) \\ \\ \dot{\beta}(t) \end{bmatrix} \qquad (3\text{-}14)$$

Here, $\alpha(t)$ and $\beta(t)$ describe the time varying azimuth and elevation velocities in the inertial reference frame. Also, $\underline{B}_D(t) = \underline{I}$ is the control input matrix. At this point, a discrete representation of (3-14) can be made over the interval $[t_i, t_{i+1}]$ in terms of the target acceleration and velocity at time t_i:

$$\underline{x}_D(t_{i+1}) = \underline{x}_D(t_i) + \dot{\underline{x}}_D(t_i)\,\Delta t + \ddot{\underline{x}}_D(t_i)\,\Delta t^2/2 \qquad (3\text{-}15)$$

which is valid to the degree that the accelerations $\underline{x}_D$ are constant over a given sample period.

At any one instant in time the geometry of the target in the inertial reference frame might appear as shown in Figure 3.

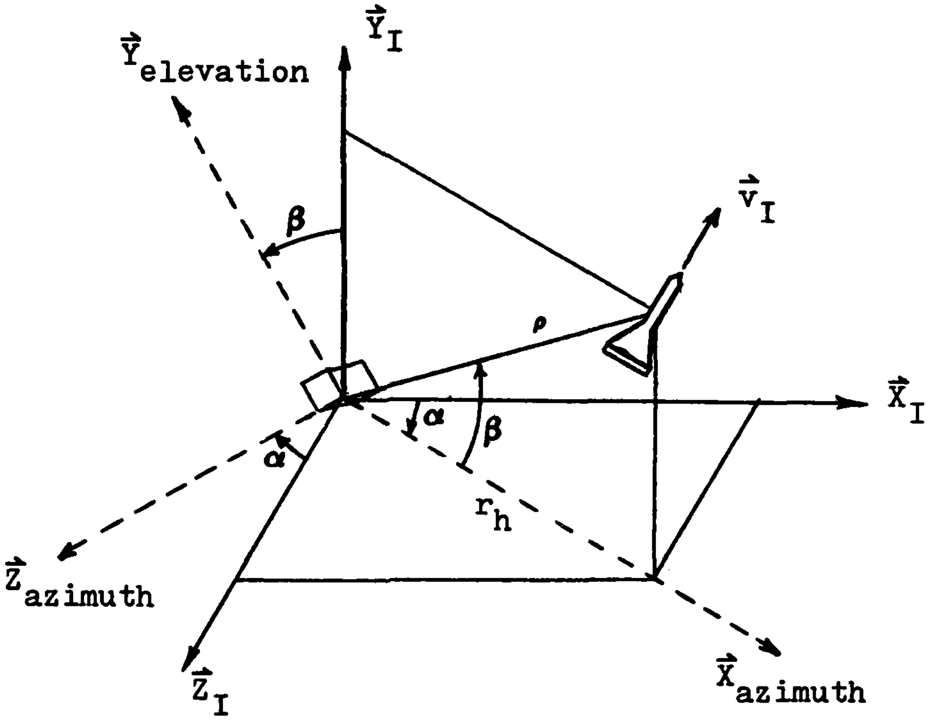


Figure 3. Inertial Reference Frame

where,

$\vec{X}_I, \vec{Y}_I, \vec{Z}_I$ = inertial axes

ρ = range to target from inertial origin

$\vec{v}_I$ inertial velocity

r_h = horizontal range

α = azimuth angular displacement from $\vec{X}_I$ about $\vec{Y}_I$

β = elevation angular displacement from the $\vec{X}_I$ - $\vec{Z}_I$ plane about the $\vec{Z}_{azimuth}$ axis of Fig 3.

In order to transform quantities to the FLIR image plane from inertial space, the azimuth and elevation geometries must first be defined in terms of inertial coordinates. Figure 4 shows a view of the azimuth geometry by looking at a projection of the target onto the horizontal plane.

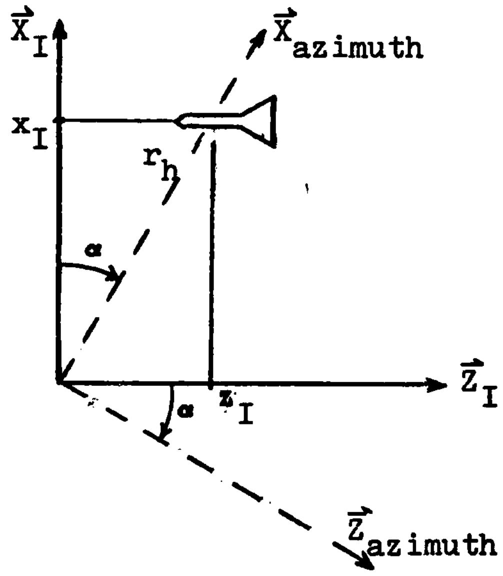


Figure 4. Azimuth Geometry

Thus the azimuth axes are displaced from the inertial $\vec{X}_I$ and $\vec{Z}_I$ axes by the angle $\alpha(t)$ defined as:

$$\alpha(t) = \tan^{-1}[z_I(t)/x_I(t)] \qquad \text{(radians)} \qquad (3\text{-}16)$$

which implies,

$$\dot{\alpha}(t) = [\dot{z}_I(t)x_I(t)-\dot{x}_I(t)z_I(t)]/r_h^2 \qquad \text{(rad/sec)} \qquad (3\text{-}17)$$

and,

$$\ddot{\alpha}(t) = [(x_I(t)\ddot{z}_I(t)-\ddot{x}_I(t)z_I(t))r_h^2-2(x_I(t)\dot{x}_I(t)$$

$$+ z_I(t)\dot{z}_I(t))(x_I(t)\dot{z}_I(t)-\dot{x}_I(t)z_I(t))]/r_h^4 \text{(rad/sec}^2\text{)} \qquad (3\text{-}18)$$

where $r_h^2 = x_I^2 + z_I^2$. The results of Eqs (3-16), (3-17), and (3-18) can be converted from units involving radians to pixels by dividing by (20 µrad/pixel) since the 8x8 pixel field of view (FOV) is assumed to represent a region of 160 µrad x 160 µrad.

Similarly, Figure 5 displays the geometry involved in calculating the elevation, elevation velocity, and elevation acceleration. The plane shown is perpendicular to the $\vec{X}_I$-$\vec{Z}_I$ plane and contains the $\vec{Y}_I$ axis, ρ, and r_h.

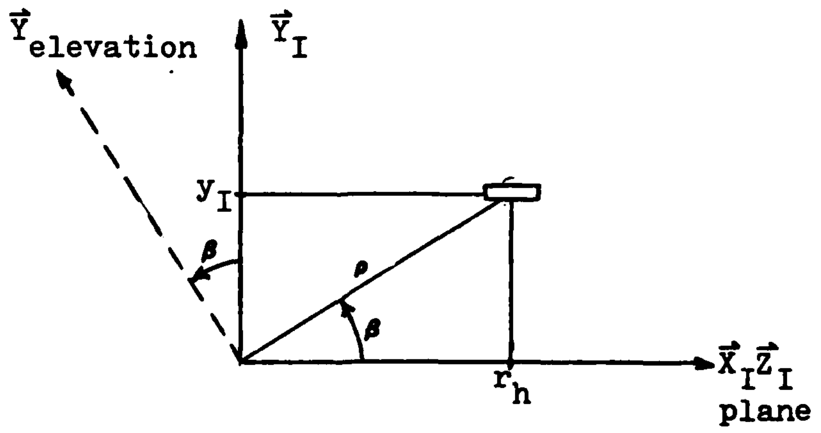


Figure 5. Elevation Geometry

$$\rho = \text{range} = [x_I^2(t) + y_I^2(t) + z_I^2(t)]^{1/2}$$

$$r_h = \text{horizontal range} = [x_I^2(t) + z_I^2(t)]^{1/2}$$

Therefore,

$$\beta(t) = \tan^{-1}[y_I(t)/r_h(t)] \qquad \text{(radians)} \quad (3\text{-}19)$$

which implies,

$$\dot{\beta}(t) = [r_h(t)\dot{y}_I(t) - y_I(t)\dot{r}_h(t)]/\rho^2(t) \qquad \text{(rad/sec)} \quad (3\text{-}20)$$

where,

$$\dot{r}_h(t) = [\dot{x}_I(t)x_I(t) + \dot{z}_I(t)z_I(t)]/r_h(t) \qquad \text{(m/sec)} \quad (3\text{-}21)$$

In addition,

$$\ddot{\beta}(t) = \{\rho^2(t)[r_h(t)\ddot{y}_I(t) - y_I(t)\ddot{r}_h(t)] - [r_h(t)\dot{y}_I(t) - y_I(t)\dot{r}_h(t)](2\dot{\rho}(t)\,\rho(t))\}/\rho^4(t) \qquad (\text{rad/sec}^2) \quad (3\text{-}22)$$

where,

$$\ddot{r}_h(t) = \{[x_I(t)\ddot{x}_I(t) + \dot{x}_I^2(t) + z_I(t)\ddot{z}_I(t) + \dot{z}_I^2(t)]r_h(t) - \dot{r}_h(t)[x_I(t)\dot{x}_I(t) + z_I(t)\dot{z}_I(t)]\}/r_h^2(t) \qquad (\text{m/sec}^2) \quad (3\text{-}23)$$

and,

$$\dot{\rho}(t) = [x_I(t)\dot{x}_I(t) + y_I(t)\dot{y}_I(t) + z_I(t)\dot{z}_I(t)]/\rho(t) \qquad \text{(m/sec)} \quad (3\text{-}24)$$

The quantities $\beta(t)$, $\dot{\beta}(t)$, and $\ddot{\beta}(t)$ can also be converted to units involving pixels by dividing by (20 rad/pixel).

Returning to Eq (3-15) the general solution of the dynamics differential equation is:

$$\underline{x}_D(t) = \underline{\Phi}_D(t,t_i)\underline{x}_D(t_i) + \int_{t_i}^{t} \underline{\Phi}_D(t,\tau)\underline{B}_D(\tau)\underline{u}_D(\tau)d\tau \tag{3-25}$$

where

t_i = initial time

$\underline{\Phi}_D(t,\tau)$ = state transition matrix for vehicle dynamics

$\underline{B}_D(\tau)$ = the control input matrix

$\underline{u}_D(\tau)$ = the control function for the truth model dynamics

$= [\,\dot{\alpha}(\tau)\ ,\ \dot{\beta}(\tau)\,]^T$

For digital computer implementation, a desirable form of $\underline{u}_D(t)$ is one that assumes piecewise constant accelerations between sampling times. If an approximation such as this is reasonable, then a discrete representation of Eq (3-25) would be a function of discrete time instants. That is,

$$\underline{x}_D(t_{i+1}) = \underline{\Phi}_D(t_{i+1},t_i)\underline{x}_D(t_i) + \underline{B}_d(t_i)\underline{u}_d(t_i) \tag{3-26}$$

which is accurate if

$$\underline{B}_d(t_i)\underline{u}_d(t_i) \cong \int_{t_i}^{t_{i+1}} \underline{\Phi}_D(t_{i+1},\tau)\underline{B}_D(\tau)\underline{u}_D(\tau)d\tau \tag{3-27}$$

Specifically, if the acceleration is assumed piecewise constant, the azimuth position due to target dynamics can be propagated as

$$\alpha(t_{i+1}) = \alpha(t_i) + \Delta t\,\dot{\alpha}(t_i) + (\Delta t^2/2)\,\ddot{\alpha}(t_i) \qquad (3\text{-}28)$$

This, along with the corresponding elevation propagation, can be used as a reasonable approximation to the integral in Eq (3-27). As a result, the discrete time control and control input matrices can be described as

$$\underline{B}_d(t_i)\underline{u}_d(t_i) = \begin{bmatrix} \Delta t & 0 & \frac{\Delta t^2}{2} & 0 \\ 0 & \Delta t & 0 & \frac{\Delta t^2}{2} \end{bmatrix} \begin{bmatrix} \dot{\alpha}(t_i) \\ \dot{\beta}(t_i) \\ \ddot{\alpha}(t_i) \\ \ddot{\beta}(t_i) \end{bmatrix} \qquad (3\text{-}29)$$

It should be noted that the discrete time history, $\underline{x}_D(t_i)$, could have been easily specified from Eqs (3-16) and (3-19); however, the quantities $\dot{\alpha}$, $\dot{\beta}$, $\ddot{\alpha}$, and $\ddot{\beta}$ were needed for filter performance analysis and when used in the manner shown above, they adapt well to a state space representation of the stochastic truth model. Of course, Eq (3-29) is not really expressed in terms of four independent inputs, since $\dot{\alpha}$ and $\ddot{\alpha}$ are interdependent, as are $\dot{\beta}$ and $\ddot{\beta}$.

Although the dynamics of the center of gravity of the target in inertial space can be specified via (t) and (t), the dynamics of other points on the craft are somewhat more complicated to describe during rolling and/or G-pulling maneuvers. Consideration of the dynamics of such points is important if a realistic multiple hot-spot image is to be created on the FLIR image plane since a target will not, in general, be sperically symmetric. In this research, the target is assumed to resemble three ellipsoids of revolution in three-dimensional space. The centers of the ellipsoids are fixed distances apart and lie in a plane containing the velocity vector of the craft, $\vec{v}_I$, as well as the semi-major axes of the ellipsoids (see Section 3.5 on FLIR Measurements for more detail).

Thus, when these ellipsoids are projected onto the FLIR image plane, bivariate Gaussian elliptical equal-intensity contours will appear with the center of each ellipsoid projecting onto the maximum intensity point of its corresponding FLIR image plane intensity distribution. In addition, the one-sigma values of the bivariate Gaussian intensity distributions will be proportional to the semi-minor and semi-major axes of the corresponding target ellipsoids. To keep track of the location of the maximum intensity points of these distributions, the following planes and frames are defined in relation to the inertial reference frame.

<u>Target plane</u> - The plane formed by the velocity vector, $\vec{v}_I$, of the target and the centers of the ellipsoids of revolution representing the target (e.g. the plane of the wings of an aircraft target). It is specified via, $\vec{e}_v$, a unit vector in the craft's velocity direction, and $\vec{e}_{pv}$, a unit vector that is perpendicular to $\vec{e}_v$ and points toward the starboard side of the craft. Defining a third unit vector, $\vec{e}_{ppv} = \vec{e}_{pv} \times \vec{e}_v$, completes the definition of the target reference frame.

$\alpha\beta$ plane - The plane containing the center of gravity of the vehicle and oriented such that it is always perpendicular to the line of sight (LOS) from the inertial origin (tracker location) to the target. It is defined by $\vec{e}_\alpha$ and $\vec{e}_\beta$. The vector $\vec{e}_\alpha$ is a unit vector in the plane that is horizontal and when translated to the inertial origin is displaced from the $\vec{Z}_I$ axis by the angle $\alpha(t)$ in the horizontal plane. The vector $\vec{e}_\beta$ is a unit vector in the $\alpha\beta$ plane that is perpendicular to $\vec{e}_\alpha$ and when translated to the inertial origin is displaced from $\vec{Y}_I$ by the angle $\beta(t)$ in the plane formed by $\vec{Y}_I$ and the LOS. Also of use when determining which ellipsoid is closest (for purposes to be seen in the discussion of Eq (3-48)) is the range unit vector $\vec{e}_r$ which lies along the LOS.

These two planes are illustrated in Figure 6.

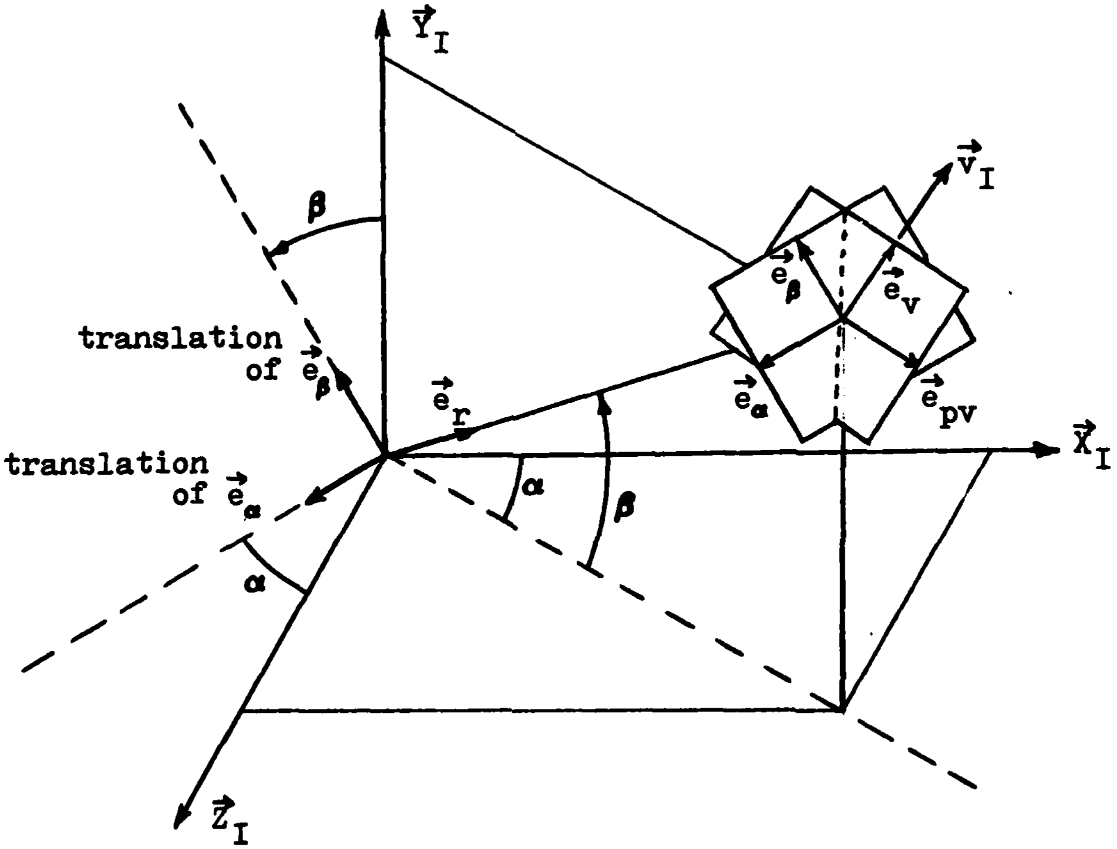


Figure 6. Target and $\alpha\beta$ planes

These particular planes are physically meaningful since projections of vectors from the target plane onto the plane can be directly related to angular displacements of alpha and beta, and thus, to pixel distances along the x and y FLIR axes. Therefore, by keeping track of the vectors from the target center of gravity (CG) to the centers of the ellipsoids, the relative positions of the maximum intensity points on the FLIR image plane can be determined.

To accomplish this, each of the unit vectors discussed above must be expressed in terms of the inertial unit vectors: $\vec{i}$, $\vec{j}$, and $\vec{k}$. This, in effect, establishes the direction cosine matrices that relate the vectors, so defined, to the inertial frame. Thus,

$$\vec{e}_\alpha = -\sin\alpha\,\vec{i} + \cos\alpha\,\vec{k}$$

so

$$e^I_\alpha = (-\sin\alpha,\ 0,\ \cos\alpha)^T \qquad (3\text{-}30a)$$

and

$$\vec{e}_\beta = -\cos\alpha\sin\beta\,\vec{i} + \cos\beta\,\vec{j} - \sin\alpha\sin\beta\,\vec{k}$$

so

$$e^I_\beta = (-\cos\alpha\sin\beta,\ \cos\beta,\ -\sin\alpha\sin\beta)^T \qquad (3\text{-}30b)$$

where the I superscript denotes "coordinatized in the I frame." The unit vectors $\vec{e}_v$, $\vec{e}_{pv}$, and $\vec{e}_{ppv}$ can also be written explicitly once the particular maneuver the craft is to perform is known. Following are typical trajectories considered in this research.

Trajectory 1: This trajectory was used to evaluate basic filter performance versus a dynamic target image on the FLIR image plane. It involves straight line propagation (from some initial inertial position) parallel to the $\vec{X}_I$ axis while undergoing a constant velocity, constant roll-rate maneuver. Specifically,

$$(x_{Io}, y_{Io}, z_{Io}) = (5000,\ 500,\ 20000) \quad (m) \qquad (3\text{-}31a)$$

$$v_I^I = v_{Io}^I = (-1000,\ 0,\ 0)^T \ (m/s) \qquad (3\text{-}31b)$$

$$a_I^I = a_{Io}^I = (0,\ 0,\ 0)^T \quad (m/s)^2 \qquad (3\text{-}31c)$$

Thus a clockwise roll will result in both the linear and angular velocity unit vectors ($\vec{e}_v$ and w/ $\vec{w}$) being oriented in the $-\vec{i}$ direction. The angle γ_1 will represent the angular displacement from a horizontal initial attitude (i.e. γ_1 is the angle from $\vec{j}$ to $\vec{e}_{ppv}$, in the $\vec{Y}_I$-$\vec{Z}_I$ plane when velocity is totally in the +/- $\vec{X}_I$ direction).

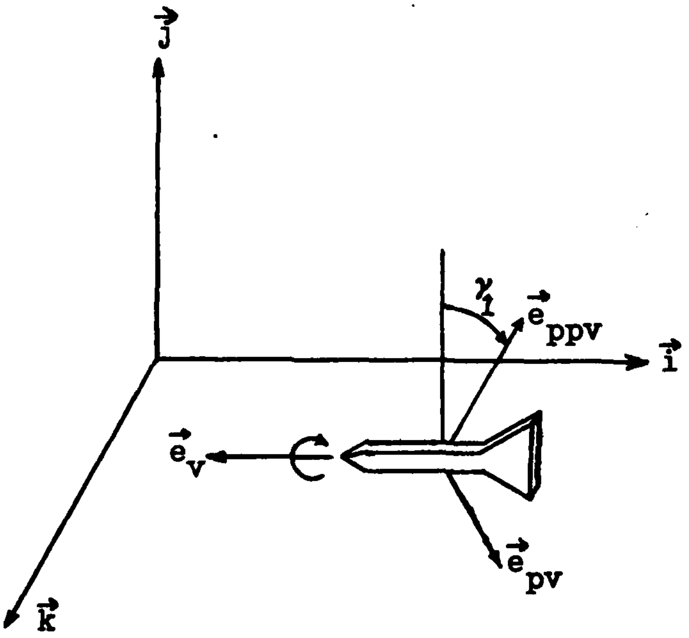


Figure 7. Trajectory 1

In this case,

$$e_v^I = (-1,\ 0,\ 0)^T \tag{3-32a}$$

$$e_{pv}^I = (0,\ -\sin\gamma_1,\ -\cos\gamma_1)^T \tag{3-32b}$$

$$e_{ppv}^I = (0,\ \cos\gamma_1\ ,\ -\sin\gamma_1)^T \tag{3-32c}$$

with $\gamma_1 = \gamma_{1o} + \omega t$ for a constant roll-rate of ω rad/sec.

Trajectory 2: In order to evaluate filter performance against more dynamic maneuvers this trajectory involves a constant G pull-up. For ease of implementation, the maneuver was conducted in a plane parallel to the $\vec{X}_I$-$\vec{Y}_I$ plane starting from horizontal flight in an initial velocity direction equal to $-\vec{i}$. An angle γ_2 represents the angular displacement from the horizontal attitude (i.e. the angle between $\vec{e}_{ppv}$ and $\vec{j}$ in the $\vec{X}_I$-$\vec{Y}_I$ plane). Prior to the maneuver, target dynamics are described by Eq (3-31). After maneuver initiation at time t', the velocity equations are described by

$$v_x = -1000 \cos[\omega'(t-t')] \tag{3-33a}$$

$$v_y = 1000 \sin[\omega'(t-t')] \tag{3-33b}$$

$$v_z = 0. \tag{3-33c}$$

where ω' is dependent on the magnitude of the G pull-up desired. The geometry involved with this maneuver is shown in Figure 8

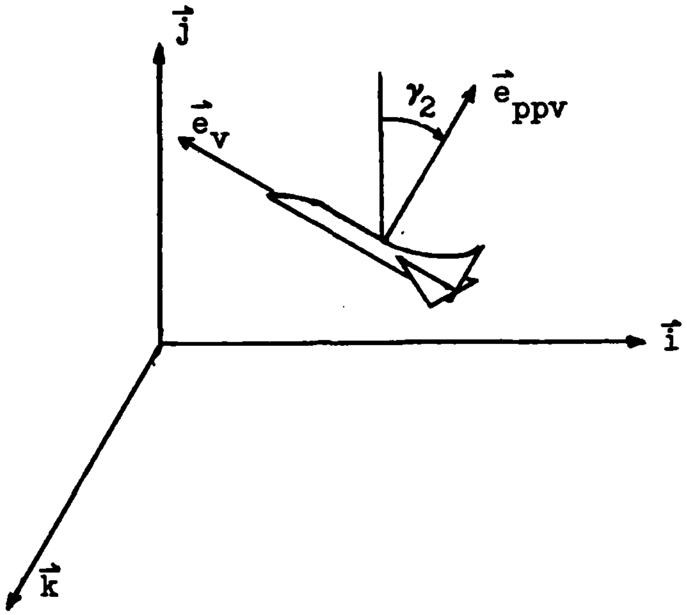


Figure 8. Trajectory 2

where,

$$e_v^I = (-\cos\gamma_2,\ \sin\gamma_2,\ 0)^T \tag{3-34a}$$

$$e_{pv}^I = (0,\ 0,\ 1)^T \tag{3-34b}$$

$$e_{ppv}^I = (\sin\gamma_2,\ \cos\gamma_2,\ 0)^T \tag{3-34c}$$

Trajectory 3: This trajectory is the general case of Trajectory 1 where the velocity vector is oriented in an arbitrary direction. The vectors, $\vec{e}_v$ and $\vec{\omega}$, are again defined to be along the velocity direction with $\vec{e}_{pv}$ again assumed initially horizontal. The variable γ_3 represents the angular displacement of the craft (in the $\vec{e}_{pv}$-$\vec{e}_{ppv}$ plane) from the horizontal orientation as the maneuver progresses.

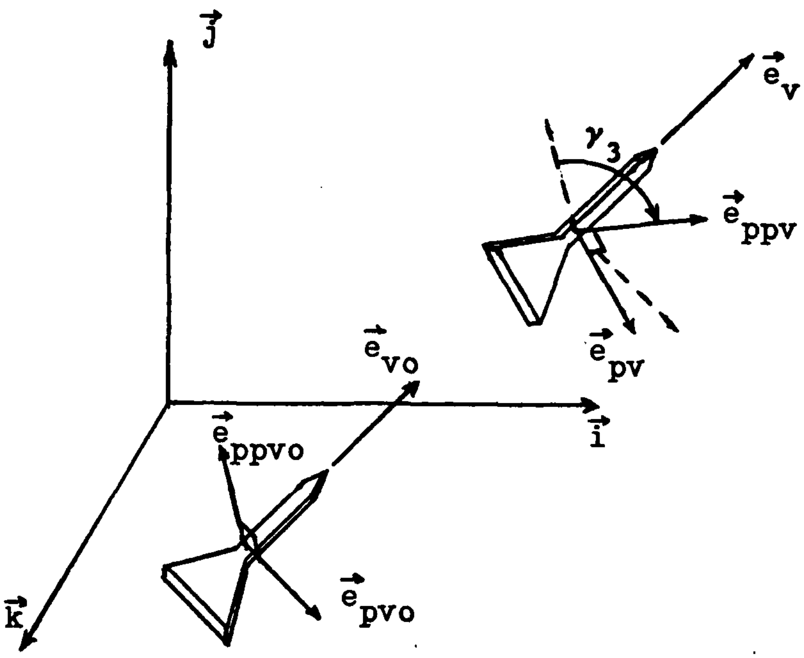


Figure 9. Trajectory 3

In this case, the target frame unit vectors are (in terms of the inertial velocity components):

$$e_v^I = e_{vo}^I = \left[\frac{v_x}{|\vec{v}_I|}, \quad \frac{v_y}{|\vec{v}_I|}, \quad \frac{v_z}{|\vec{v}_I|} \right]^T \tag{3-35}$$

and since $\vec{e}_{pv}$ is assumed initially horizontal,

$$\vec{e}_{pvo} = \frac{\vec{e}_v \times \vec{j}}{|\vec{e}_v \times \vec{j}|} = \frac{\vec{v}_I \times \vec{j}}{|\vec{v}_I \times \vec{j}|} \tag{3-36}$$

where the denominator term is required, in general, to normalize the vector resulting from the numerator cross product. Thus, using the determinant expansion method to evaluate the cross product,

$$\vec{v}_I \times \vec{j} = \begin{bmatrix} \vec{i} & \vec{j} & \vec{k} \\ v_x & v_y & v_z \\ 0 & 1 & 0 \end{bmatrix} = -v_z \vec{i} + v_x \vec{k} \qquad (3\text{-}37)$$

then,

$${}^{I}e_{pvo} = [-v_z/v_x^2 + v_z^2)^{1/2} , 0, v_x/(v_x^2 + v_z^2)^{1/2}]^T \qquad (3\text{-}38)$$

Similarly,

$$\vec{e}_{ppvo} = \vec{e}_{pvo} \times \vec{e}_{vo} = \frac{(\vec{v}_I \times \vec{j}) \times \vec{v}_I}{|(\vec{v}_I \times \vec{j}) \times \vec{v}_I|} = \frac{\vec{A}}{|\vec{A}|} \qquad (3\text{-}39)$$

Using the determinant expansion method along with Eq (3-37) to determine A results in

$$\vec{A} = (\vec{v}_I \times \vec{j}) \times \vec{v}_I = \begin{bmatrix} \vec{i} & \vec{j} & \vec{k} \\ -v_z & 0 & v_x \\ v_x & v_y & v_z \end{bmatrix}$$

$$= -v_x v_y \vec{i} + (v_x^2 + v_z^2)\vec{j} - v_y v_z \vec{k}$$

where

$$|\vec{A}| = [(v_x v_y)^2 + (v_x^2 + v_z^2) + (v_y v_z)^2]^{1/2}$$

$$= [(v_x^2 + v_z^2)(v_x^2 + v_y^2 + v_z^2)]^{1/2} \qquad (3\text{-}41)$$

and finally, the vector

$$\vec{e}_{ppvo} = (v_x^2+v_z^2)(v_x^2+v_y^2+v_z^2)^{-1/2}[-v_xv_y\vec{i} + (v_x^2+v_z^2)\vec{j} - v_yv_z\vec{k}] \quad (3\text{-}43)$$

Therefore, for a constant roll-rate about $\vec{e}_v = \vec{e}_{vo}$,

$$\vec{e}_{pv} = \cos\gamma_3\, \vec{e}_{pvo} - \sin\gamma_3\, \vec{e}_{ppvo} \quad (3\text{-}43a)$$

and

$$\vec{e}_{ppv} = \sin\gamma_3\, \vec{e}_{pvo} + \cos\gamma_3\, \vec{e}_{ppvo} \quad (3\text{-}43b)$$

which can then be related to $\vec{i}$, $\vec{j}$, and $\vec{k}$ via Eqs (3-38) and (3-42).

In all three trajectories described above γ_1, γ_2, and γ_3 can all be specified as functions of time depending on the speed of the roll or magnitude of the G pull-up that is desired. As a result, a piecewise continuous trajectory can be formed with combinations of maneuvers such as these. Use of the target frame during such maneuvers can be illustrated by examining the target geometry shown in Fig 10.

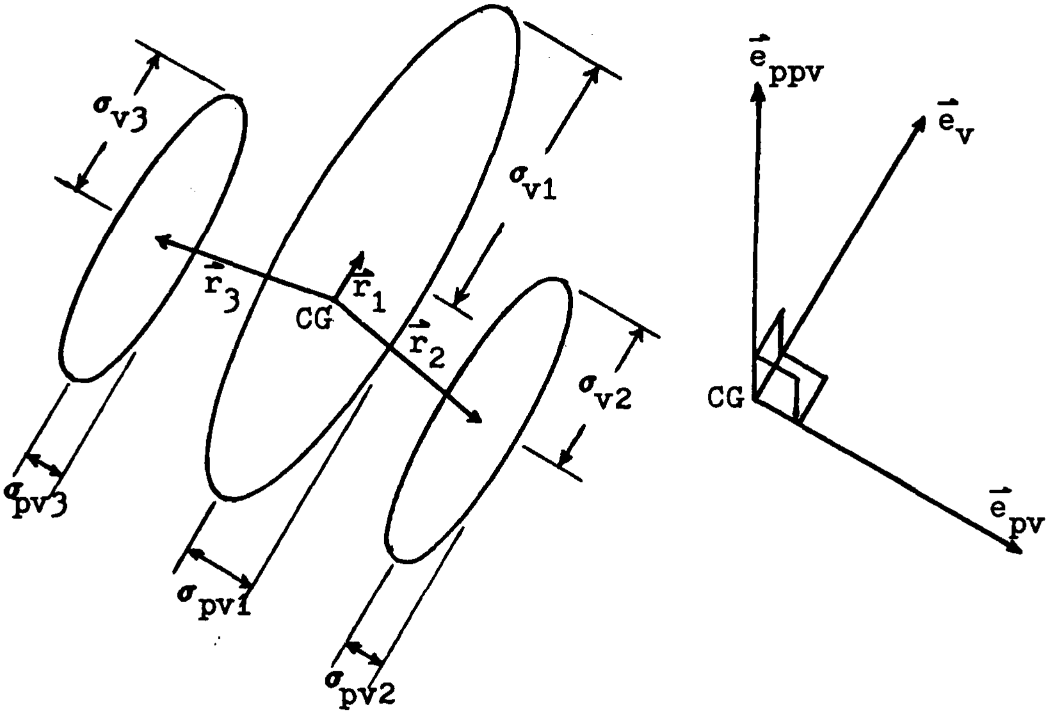


Figure 10. Target Geometry

The position of the center of each ellipsoid of revolution can be specified in target coordinates (relative to CG position) as

$$\vec{r}_m = \delta_{v_m} \vec{e}_v + \delta_{pv_m} \vec{e}_{pv} \qquad (m) \qquad (3\text{-}44)$$

for m = 1,2, and 3. The parameters δ_{v_m} and δ_{pv_m} represent physical distances (meters) of the kth ellipsoid from the CG in the velocity direction and direction perpendicular to the velocity respectively. This vector can then be projected onto the $\alpha\beta$ plane to produce δ_{α_m} and δ_{β_m} (also in meters).

These values can then be related to angular displacements of α and β as $\Delta\alpha_m$ and $\Delta\beta_m$ respectively.

Since the angular displacements will be small compared to the target range (ρ), the small angle approximation yields:

$$\Delta\alpha_m = \tan(\Delta\alpha_m) = \delta_{\alpha_m}/\rho \tag{3-45a}$$

$$\Delta\beta_m = \tan(\Delta\beta_m) = \delta_{\beta_m}/\rho \tag{3-45b}$$

where

$$\delta_{\alpha m} = \vec{r}_m \cdot \vec{e}_\alpha \tag{3-46a}$$

$$\delta_{\beta m} = r_m \cdot e_\beta \tag{3-46b}$$

Inserting Eqs (3-44) and (3-46) into Eq (3-45) and converting radians to pixels produces

$$\Delta\alpha_m = (20\text{x}10^{-6})^{-1}[\delta_{v_m} \vec{e}_v \cdot \vec{e}_\alpha + \delta_{pv_m} \vec{e}_{pv} \cdot \vec{e}_\alpha] \quad \text{(pixels)} \tag{3-47a}$$

$$\Delta\beta_m = (20\text{x}10^{-6})^{-1}[\delta_{v_m} \vec{e}_v \cdot \vec{e}_\beta + \delta_{pv_m} \vec{e}_{pv} \cdot \vec{e}_\beta] \quad \text{(pixels)} \tag{3-47b}$$

which are to be used to determine the separations of the intensity peaks from the centroid location on the FLIR image plane.

One characteristic that realistic targets may exhibit on the FLIR plane is the "disappearance" of various hot-spots as the target itself obscures sight of the corresponding ellipsoids. Thus, to incorporate this feature into the simulation of a dynamic image, information must be generated on the relative ranges of the target ellipsoids. This can be done by forming

$$\Delta r_m = \vec{r}_m \cdot \vec{e}_r \qquad (m) \qquad (3\text{-}48)$$

for m = 1, 2, and 3. The smallest Δr_m will indicate which ellipsoid is closest to the tracker. These values will be used when the corresponding one-sigma images of the Gaussian distributions overlap on the image plane. This is done in order to determine which distribution will contribute to the measurement at those points.

3.4 Overall State Space Model

Augmenting the truth model target dynamics and atmospheric distortion models in both directions on the FLIR plane yields a single eight-state model which can be described by the following differential equation.

$$\dot{\underline{x}}_T(t) = \underline{F}_T\underline{x}_T(t) + \underline{B}_T\underline{u}_T(t) + \underline{G}_T\underline{w}_T(t) \tag{3-49}$$

$$\underline{y}_T(t) = \underline{H}_T\underline{x}_T(t) \tag{3-50}$$

where

$$\underline{x}_T(t) = \begin{bmatrix} x_D\ (t) \\ y_D\ (t) \\ x_{1A}(t) \\ x_{2A}(t) \\ x_{3A}(t) \\ y_{1A}(t) \\ y_{2A}(t) \\ y_{3A}(t) \end{bmatrix} \qquad \underline{F}_T = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -A & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -B & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -B & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -A & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -B & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -B \end{bmatrix}$$

$$\underline{B}_T = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad \underline{u}_T(t) = \begin{bmatrix} \dot{\alpha}(t) \\ \dot{\beta}(t) \end{bmatrix} \qquad \underline{w}_T(t) = \begin{bmatrix} w_{A1}(t) \\ w_{A2}(t) \end{bmatrix} \qquad \underline{G}_T = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ G1 & 0 \\ G2 & 0 \\ G3 & 0 \\ 0 & G1 \\ 0 & G2 \\ 0 & G3 \end{bmatrix}$$

and

$$\underline{H}_T = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

where $E[\underline{w}_T(t)] = \underline{0}$ and

$$E[\underline{w}_T(t)\underline{w}_T(t+\tau)] = \underline{Q}_T\delta(\tau) = \begin{bmatrix} Q_A & 0 \\ 0 & Q_A \end{bmatrix} \delta(\tau)$$

Thus, the equivalent discrete-time model of the overall system is described by the propagation of Eq (3-49) from time t_i to time t_{i+1} yielding

$$\underline{x}_T(t_{i+1}) = \underline{\Phi}_T(t_{i+1},t_i)\underline{x}_T(t_i) + \underline{B}_d\underline{u}_d(t_i) + \underline{G}_d\underline{w}_d(t_i) \qquad (3\text{-}51)$$

where

$$\dot{\underline{\Phi}}_T(t,t_i) = \underline{F}_T\underline{\Phi}_T(t,t_i)$$

and

$$\underline{\Phi}_T(t_i,t_i) = \underline{I}$$

Thus,

$$\underline{\Phi}_T(t_{i+1},t_i)=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & e^{-A\Delta t} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & e^{-B\Delta t} & \Delta t e^{-B\Delta t} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & e^{-B\Delta t} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & e^{-A\Delta t} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & e^{-B\Delta t} & \Delta t e^{-B\Delta t} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{-B\Delta t} \end{bmatrix}$$

and using the approximations to the azimuth and elevation velocities given in Eq (3-28)

$$\underline{B}_d = \begin{bmatrix} \Delta t & 0 & \Delta t^2/2 & 0 \\ 0 & \Delta t & 0 & \Delta t^2/2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\underline{u}_d(t_i) = [\dot{\alpha}(t_i)\ \dot{\beta}(t_i)\ \ddot{\alpha}(t_i)\ \ddot{\beta}(t_i)]^T$$

$$\underline{G}_d = {}^{c}\sqrt{\underline{Q}_{TD}} \quad \text{such that} \quad {}^{c}\sqrt{\underline{Q}_{Td}}\ {}^{c}\sqrt{\underline{Q}_{Td}}^{\,T} = \underline{Q}_{Td}$$

$$E[\underline{w}_d(t_i)] = \underline{0}$$

$$E[\underline{w}_d(t_i)\underline{w}_d(t_j)] = \underline{I}\delta_{ij}$$

and where

$$\underline{Q}_{Td} = \int_{t_i}^{t_{i+1}} \underline{\Phi}_T(t_{i+1},\tau)\underline{G}_T\underline{Q}_T\underline{G}_T^T\underline{\Phi}_T^T(t_{i+1},\tau)d\tau \qquad (3\text{-}52)$$

The details for determining the elements of $\underline{Q}_{Td}$ via exact integration are shown in Appendix A.

3.5 FLIR Measurements

The FLIR measurements are "outputs" representing the average intensity of target radiation incident on an array of detector surfaces. These values are corrupted by background and device noises. In spite of the noises present, the EKF processes a history of FLIR measurement data frames in order to determine the underlying shape of the target image; and subsequently, to predict the desired pointing direction of the high energy laser. However, the shape itself may display different characteristics depending on several factors. Specifically, the amount of target detail that can be distinguished by the FLIR is dependent upon both the atmospheric distance that the target radiation must propagate through to reach the sensor, as well as the diffraction effects inherent in the optical system. Since these factors tend to make detail less distinguishable, FLIR images of distant targets can be modelled as bivariate Gaussian intensity distributions with circular (Ref: 12) and/or elliptical (Ref: 5) equal-intensity contours. However, such models are inadequate in many cases when targets are larger or much closer since the image begins to take on more detail. Such a "multiple hot-spot" target image can be better modelled as the sum of contributions from M such bivariate Gaussian intensity distributions. Thus, the intensity output of any one pixel (picture element), within the 8x8 pixel array that constitutes one data frame, can be described as follows:

$$z_{kl}(t_i) = (1/A_p) \int_{\substack{kl\text{-th}\\ \text{pixel}}} \sum_{m=1}^{M} I_m(x,y,x_{pm}(t_i),y_{pm}(t_i))dxdy + v_{kl}(t_i) \qquad (3\text{-}53)$$

where

$I_m(\cdot)$ = the intensity of the m-th component of the target radiation on the FLIR image plane as a function of the position in the frame and the position of the peak of the mk-th hot-spot, out of a total of M hot-spots

$z_{kl}(t_i)$ = intensity output of the kl-th pixel at time t_i.

A_p = area of a single pixel.

(x,y) = the coordinates of an arbitrary point within the kl-th pixel.

$(x_{pm}(t_i),y_{pm}(t_i))$ = coordinates of the intensity peak of the m-th image hot-spot.

$v_{kl}(t_i)$ = noise added to the kl-th pixel representing FLIR and spatially correlated background noises.

When the general form of a target can be well represented by M ellipsoids of revolution in inertial space, then the projection of the target's image onto the FLIR plane can be described, in general, as M or fewer elliptically-shaped contours. In such a case, the contribution to the (x,y) point in the kl-th pixel from the bivariate Gaussian representing the m-th hot-spot can be described as

$$I_m\{\cdot\} = (Imax_m)\exp\{-.5[(x-x_{pm})(y-y_{pm})] \\ [\underline{P}_m]^{-1}[(x-x_{pm})(y-y_{pm})]^T\} \qquad (3\text{-}54)$$

where

$Imax_m$ = the maximum intensity of the m-th hot-spot.

(x_{pm}, y_{pm}) gives the location of the peak of the m-th hot spot.

$\underline{P}_m$ = the matrix correspondint to the m-th hot-spot whose eigenvalues are σ_v^2 and σ_{pv}^2. The latter two parameters represent the dispersions of the m-th bivariate Gaussian elliptical contour intensity distribution in the target velocity direction and the direction perpendicular to velocity on the FLIR image plane.

3.6 Target Image

It is assumed in this research that side-slip angle and target angle of attack are essentially zero. Thus the semi-major axes of the target ellipsoids will be aligned parallel to the inertial velocity vector. This implies that the semi-major axes of the elliptical equal-intensity contours on the FLIR plane will be aligned parallel to the FLIR velocity vector as well. A typical FLIR image of a three hot-spot target might appear as shown in Fig 11.

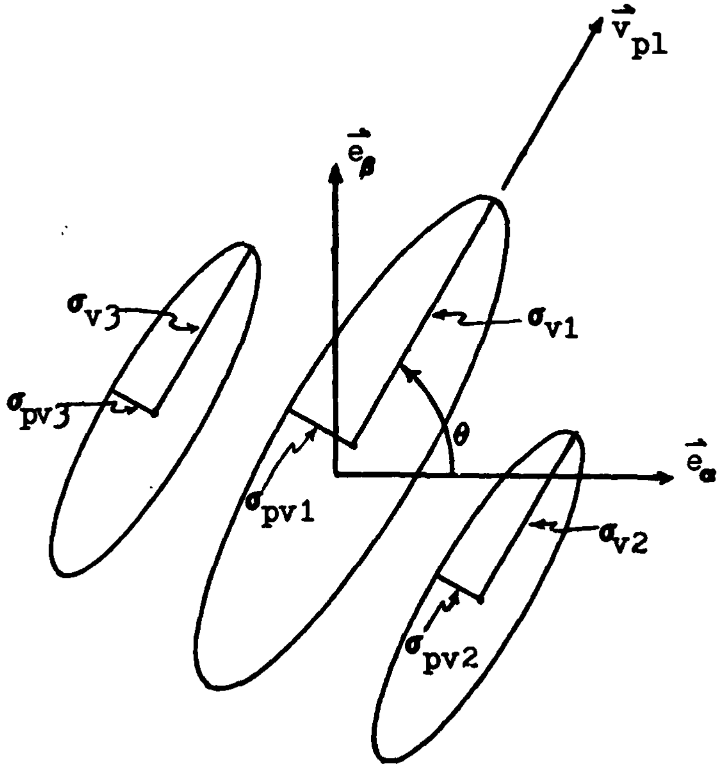


Fig 11. FLIR Image Geometry

where

$\vec{v}_{pl} = \dot{\alpha}\vec{e}_{\alpha} + \dot{\beta}\vec{e}$ = FLIR plane velocity vector (projection of $v_{\perp LOS}$ onto the FLIR plane)

$\vec{v}_{pl} = [\dot{\alpha}^2 + \dot{\beta}^2]^{1/2}$ = magnitude of the FLIR plane velocity vector (pixels/sec)

= image dispersion of the m-th distribution in the planar velocity direction.

= image dispersion of the m-th distribution in the direction perpendicular to the planar velocity.

Thus,

$$\cos\theta = \frac{\dot{\alpha}}{|\vec{v}_{pl}|} = \frac{\text{azimuth velocity}}{|\vec{v}\ \perp LOS|}$$

and

$$\sin\theta = \frac{\beta}{|\vec{v}_{pl}|} = \frac{\text{elevation velocity}}{|\vec{v}\perp LOS|} \qquad (3\text{-}55)$$

Although the target dimensions are constant, its image's dimensions vary depending on range and orientation. However, the image dimension can be specified in term of some initial size (σ_{vo_m}, σ_{pvo_m}), some initial range (ζ_o), and the current range and velocity. The initial size parameters, σ_{vo_m} and σ_{pvo_m}, are assumed specified with the target lying in the plane perpendicular to the LOS since this orientation will then produce the maximum sized image for that particular range.

Since the target is modelled as M=3 ellipsoids of revolution, the semi-minor axes of the target image, σ_{pv_m} for m = 1, 2, and 3, will only vary with a dependence on target range. Specifically,

$$\sigma_{pv} = \sigma_{pvo}\ \rho_o/\rho \qquad (3\text{-}57)$$

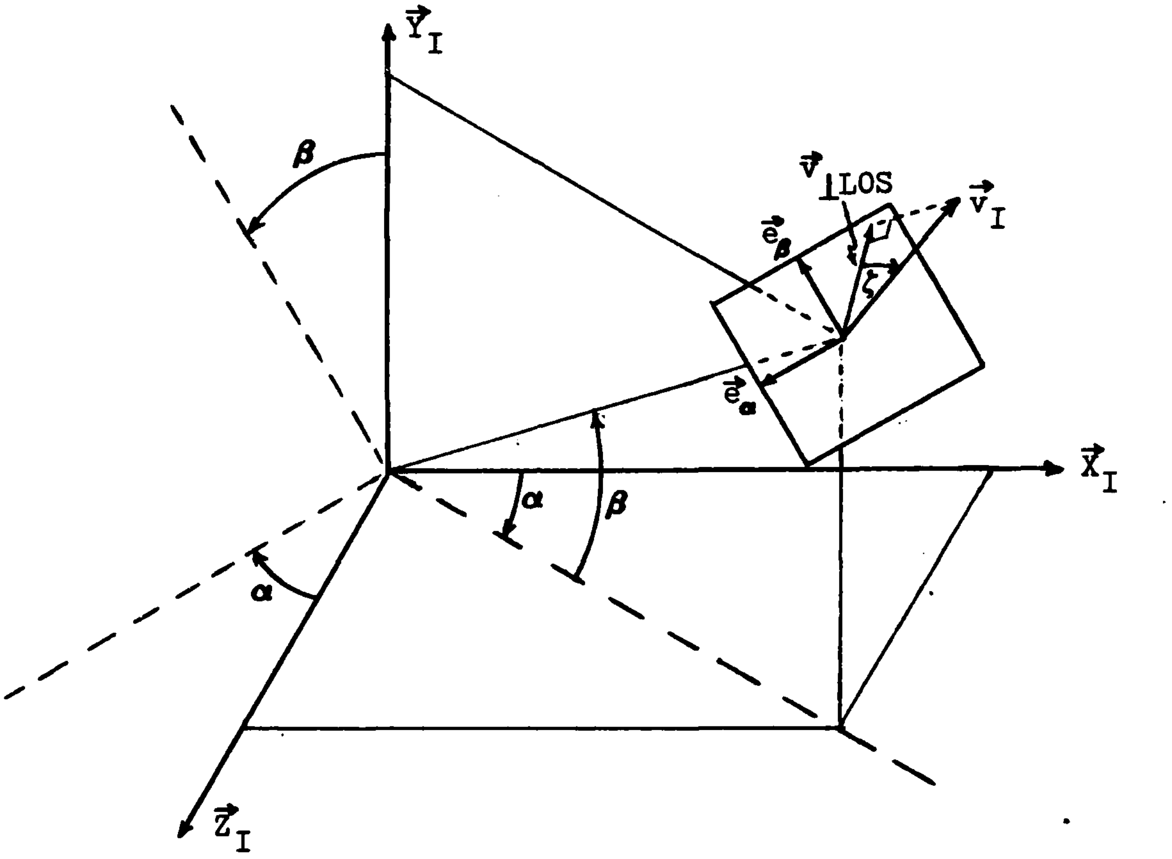


Figure 12. Plane Perpendicular to the LOS

In contrast, the semi-major axes of the target image, expressed by σ_{v_m} for m = 1, 2, and 3, will also depend on the orientation of the target with respect to the plane perpendicular to the LOS. As indicated in Fig 12 and Eq (3-58), this dependence can be expressed in terms of ζ, the angle between the inertial velocity vector and the plane that is perpendicular to the line of sight.

$$v_m = (\phi_o/\rho)[\sigma_{pvo_m} + \cos\zeta(\sigma_{vo_m} - \sigma_{pvo_m})]$$

$$= \sigma_{pv_m}[1 + (\vec{v}_{\perp LOS} / \vec{v}_I)(AR_m - 1)] \qquad (3\text{-}58)$$

where

$AR_m = \sigma_{vo_m}/\sigma_{pvo_m}$ = Maximum Aspect Ratio of the m-th hot-spot of the target image

The validity of Eq (3-58) can be demonstrated by noting that when the target is flying in the plane perpendicular to the LOS ($\zeta = 0^o$), the maximum aspect ratio is presented to the tracker with the image semi-major axes dependent on range. Specifically,

$$\sigma_{v_m} = \sigma_{vo_m} \rho_o/\rho \quad \text{for} \quad \zeta = 0^o \qquad (3\text{-}59)$$

Also, when the target is flying perpendicular to this plane ($\zeta = 90^o$), the minimum aspect ratio (AR = 1) is presented to the tracker, creating circular equal-intensity contours on the image plane. That is,

$$\sigma_{v_m} = \sigma_{pvo_m} \rho_o/\rho = \sigma_{pv_m} \quad \text{for} \quad \zeta = 90^o \qquad (3\text{-}60)$$

In reference to Figure 11, the transformation to transform vectors on the FLIR image plane that are in a frame aligned with $\vec{v}_{PL}$ (coordinates x',y') to the $\alpha\beta$ frame (coordinates x, y) is the following.

$$\underline{x} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} = \underline{A} \begin{bmatrix} x' \\ y' \end{bmatrix} = \underline{A}\,\underline{x}' \quad (3\text{-}61)$$

Transformation of a 2x2 dispersion matrix, $\underline{P}'$, from the (x',y') frame to the (x,y) frame is accomplished by

$$\underline{P} = \underline{A}\underline{P}'\underline{A}^T \qquad (3\text{-}62)$$

To compute measurement information, it is desired to transform diagonal 2x2 matrices whose eigenvalues are the inverse variances (dispersions) of the intensity distributions as expressed in the (x',y') FLIR frame. In general, the result of the transformation will no longer be diagonal in the (x,y) FLIR frame. For the direction cosine matrices relating the frames, it is easily verified that

$$\underline{A}^T = \underline{A}^{-1} \qquad (3\text{-}63)$$

Therefore, $\underline{P}'^{-1}$ will transform in the same manner as $\underline{P}'$. That is,

$$\underline{P}^{-1} = \underline{A}(\underline{P}')^{-1}\underline{A}^T \qquad (3\text{-}64)$$

To check this, $\underline{P}\underline{P}^{-1}$ can be shown to be equal to $\underline{I}$ using Eqs (3-62), (3-63), and (3-64).

$$\begin{aligned} \underline{P}\underline{P}^{-1} &= ((\underline{A}\underline{P}'\underline{A}^T)(\underline{A}(\underline{P}')^{-1}\underline{A}^T) \\ &= \underline{A}\underline{P}'\underline{I}(\underline{P}')^{-1}\underline{A}^T \\ &= \underline{A}\underline{I}\underline{A}^T \\ &= \underline{I} \qquad (3\text{-}65) \end{aligned}$$

3.7 Spatially Correlated Background Noise

In his research, Mercier (Ref: 12) modelled dicturbances in measurement information due to the FLIR and background as spatially uncorrelated, zero-mean, white, Gaussian noises. Subsequently, observations by AFWL personnel and an analysis of actual data (Ref: 5) have indicated the existence of significant spatial correlations of the background within the FLIR's field of view for "distances" of up to 40 microradians (μrad); or equivalently, two pixels on the FLIR image plane.

Modelling of spatial noise correlations in the 8x8 FLIR field of view using the measurement equation,

$$\underline{z}(t_i) = \underline{h}[\underline{x}(t_i),t_i] + \underline{v}(t_i) \tag{3-66}$$

is accomplished by allowing non-zero cross-correlations between the 64 components of the noise vector, $\underline{v}(t_i)$. Assuming the variances of the 64 background noises are of equal strength, Eq (3-66) implies the existence of a general noise covariance matrix of the form:

$$E[\underline{v}(t_i)\underline{v}^T(t_j)] = \underline{R}\delta_{ij} \tag{3-67}$$

Such that in this case,

$$\underline{R} = \sigma_b^2 \underline{R} = \sigma_b^2 \begin{bmatrix} 1 & r_{1,2} & r_{1,3} & \cdots & r_{1,64} \\ r_{2,1} & 1 & r_{1,3} & \cdots & r_{2,64} \\ r_{3,1} & r_{3,2} & 1 & \cdots & r_{3,64} \\ \vdots & \vdots & & \ddots & \vdots \\ r_{64,1} & r_{64,2} & r_{64,3} & \cdots & 1 \end{bmatrix} \qquad (3\text{-}68)$$

where

$\underline{v}(t_i)$ = 64 element spatially correlated noise vector

$\underline{R}$ = 64x64 element noise covariance matrix

$\underline{R}'$ = 64x64 element noise correlation coefficient matrix

r_{kl} = correlation coefficient representing the correlation between the scalar noise components; v_k,k=1,64 and v_l, l=1,64 at time t_i.

σ_b^2 = variance of any of the 64 background noises

More specifically, each component of the 64 element noise vector; $v_k(t_i)$ k=1,64; corresponds to the noise corrupting the k-th pixel within the 8x8 FLIR array at time t_i. The location of the k-th pixel in the FOV is defined using the following pixel numbering scheme.

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	32	32
33	34	35	36	37	38	38	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

Figure 13. Pixel Numbering Scheme

Therfore, modelling non-zero spatial correlations between the k-th pixel and all other pixels within a two pixel distance results in first and second nearest neighbor correlations. The non-zero terms in the noise covariance matrix due to such correlations with the k-th pixel of Figure 13 are indicated in Figure 14. The term in the upper-left portion of each pixel displayed in Figure 14 corresponds to the pixel location in the numbering scheme shown in Figure 13. Each subscripted term in Figure 14 represents the correlation coefficient relating the spatial correlation between that particular pixel and the k-th pixel. For zero-mean random variables the statistical definition of the correlation coefficient is (Ref: 7, p 91):

k-18 $r_{k,k-18}$	k-17 $r_{k,k-17}$	k-16 $r_{k,k-16}$	k-15 $r_{k,k-15}$	k-14 $r_{k,k-14}$
k-10 $r_{k,k-10}$	k-9 $r_{k,k-9}$	k-8 $r_{k,k-8}$	k-7 $r_{k,k-7}$	k-6 $r_{k,k-6}$
k-2 $r_{k,k-2}$	k-1 $r_{k,k-1}$	k $r_{k,k}$	k+1 $r_{k,k+1}$	k+2 $r_{k,k+2}$
k+6 $r_{k,k+6}$	k+7 $r_{k,k+7}$	k+8 $r_{k,k+8}$	k+9 $r_{k,k+9}$	k+10 $r_{k,k+10}$
k+14 $r_{k,k+14}$	k+15 $r_{k,k+15}$	k+16 $r_{k,k+16}$	k+17 $r_{k,k+17}$	k+18 $r_{k,k+18}$

Fig 14. Non-zero Correlations with the k-th Pixel

$$r_{k,l} = \frac{E[v_k(t_i)v_l(t_i)]}{\sqrt{E[v_k^2(t_i)]E[v_l^2(t_i)]}} = \frac{E[v_k(t_i)v_l(t_i)]}{\sigma_k \sigma_l} \qquad (3\text{-}69)$$

The results of the analysis of real noise data (Ref: 5) indicate that spatial correlation can be reasonably modelled as an exponentially decaying, radially symmetric function of the distance between pixels. Thus, the non-zero correlation coefficients ($r_{k,l}$) can be generated using

$$r_{k,l} = \exp(-d_{k,l}) \qquad (3\text{-}70)$$

where $d_{k,l}$ is the distance in pixels from the center of the k-th pixel to the center of the l-th pixel. From Eq (3-70) it is reasonable that correlations beyond the second nearest neighbor be ignored since, for pixels beyond these second nearest neighbors, the corresponding correlation coefficient is over an order of magnitude smaller than $r_{k,k} = 1.0$. From Eqs (3-69) and (3-70) it can be seen that $r_{k,l} = r_{l,k}$ and that, for the first and second nearest neighbors, the pixel distances and resulting correlation coefficients are as indicated in Table 3.1. Upon examining the entries in Table 3.1 and their corresponding locations as indicated by Figure 14; it can be seen that the first coefficient listed in Table 3.1 would appear once, the second, third, fourth, and sixth would appear four times while the fifth entry would appear eight times. All other correlations with the k-th pixel would be zero. Thus only six different correlation coefficients are needed to model spatial correlations using the second nearest neighbor restriction. The appropriate values of the noise covariance matrix are determined by multiplying the components of the correlation coefficient matrix by the desired noise variance as indicated in Eq (3-68). Realizations of the noise vector, $\underline{v}(t_i)$, can be simulated by premultiplying $\underline{v}'(t_i)$, a vector of unit variance, spatially uncorrelated, white, Gaussian noise, by $\sqrt[c]{\mathbf{R}}$ the Cholesky square root of the covariance matrix.

Table 3.1. Non-zero Correlation Coefficients

Pixel Distance d_{kl}	Correlation Coefficient r_{kl}
0.0	1.0
1.0	.3679
$\sqrt{2}$	.2431
2.0	.1353
$\sqrt{5}$	.1068
$\sqrt{8}$	.0591

Thus

$$\underline{v}(t_i) = \sqrt{\underline{R}}^{c}\,\underline{v}'(t_i) \tag{3-71}$$

and

$$\sqrt{\underline{R}}^{c}\,\sqrt{\underline{R}}^{c\,T} = \underline{R} \tag{3-72}$$

where

$v'(t_i)$ = a 64 component vector of zero-mean, unit variance, white, independent components (i.e. $E[\underline{v}'(t_i)\underline{v}'(t_j)] = \underline{I}\delta_{ij}$)

$\overset{c}{\sqrt{\mathbf{R}}}$ = Lower triangular Cholesky square root of the covariance matrix.

The following illustrates that the covariance matrix of the noise generated from this method is equivalent to that defined in Eq (3-67).

$$
\begin{aligned}
E[\underline{v}(t_i)\underline{v}(t_j)] &= E[\overset{c}{\sqrt{\mathbf{R}}}\underline{v}'(t_i)\underline{v}'(t_j)^T \; \overset{c}{\sqrt{\mathbf{R}}}^T] \\
&= \overset{c}{\sqrt{\mathbf{R}}} \, E[\underline{v}'(t_i)\underline{v}'(t_j)^T] \; \overset{c}{\sqrt{\mathbf{R}}}^T \\
&= \overset{c}{\sqrt{\mathbf{R}}} \, \mathbf{I} \; \overset{c}{\sqrt{\mathbf{R}}} \; \delta_{ij} \\
&= \mathbf{R} \, \delta_{ij} \qquad (3\text{-}73)
\end{aligned}
$$

A description of how the realizations of the vector, $\underline{v}'(t)$, are obtained through the use of pseudorandom codes can be found in Appendix C.

3.9 Summary

This chapter presented the mathematical models used to represent the real world. These models described the effects of target dynamics, atmospheric jitter, and spatially correlated background noises on the location of the target's intensity distribution on the FLIR image plane. Once defined, the atmospheric and dynamics models were combined in a single state space model for ease in propagating the states through time. Detailed consideration was also given to creating the measurement data required in the computer simulation. Efforts were made to locate the hot-spots of multiple hot-spot images in a dynamic manner as target maneuvers progress.

IV. Extended Kalman Filters

4.1 Introduction

In principle, use of an extended Kalman filter in this application assumes that the nonlinear intensity function within Eq (3.53) can be linearized about $\hat{x}(t_i)$ by using a first order Taylor series approximation, at least for filter gain calculation. Although the validity of this assumption is dependent upon the degree of nonlinearity involved in the state dynamics, the high measurement rate used in this research indicates, comparatively, that such an approximation is not inappropriate. This chapter presents the development of the two different dynamics models as well as the atmospheric jitter model used by the extended Kalman filters in this research. Both dynamics models estimate position, velocity, and acceleration of the target image on the FLIR plane. However, the first model discussed represents the target acceleration as a first order Gauss-Markov process. In contrast, the second dynamics model portrays the acceleration as that of a constant speed, constant turn-rate process. Since the dynamics of the filters based on these models are different, they employ different state propagation equations; however, both use the same set of measurement update equations as presented in the final portion of this chapter.

4.2 First Order Gauss-Markov Acceleration Model

In this section the target acceleration and atmospheric jitter position states are modelled as stationary, first order, Gauss-Markov (GM) processes. Representations of each such process can be generated as the output of a first order lag driven by zero-mean white Gaussian noise. It is believed that a first order model for filter estimation of atmospheric jitter is sufficient; in contrast to the third order atmospheric truth model. Since the double pole in the truth model is sufficiently isolated in the left-half s-plane, neglecting it does not seriously affect the lower frequency characteristics of the jitter. Thus the state vector can be written as

$$\underline{x}_F = [x_D, y_D, v_x, v_y, a_x, a_y, x_A, y_A]^T \tag{4-1}$$

or in terms of two-dimensional vectors

$$\dot{\underline{x}}_D = \underline{v} \tag{4-2a}$$

$$\dot{\underline{v}} = \underline{a} \tag{4-2b}$$

$$\dot{\underline{a}} = (-1/\tau_{DF})\underline{a} + \underline{w}_D \tag{4-2c}$$

$$\dot{\underline{x}}_A = (-1/\tau_{AF})\underline{x}_A + \underline{w}_A \tag{4-2d}$$

and finally,

$$\dot{\underline{x}}_F(t) = \underline{F}_F \underline{x}_F(t) + \underline{G}_F \underline{w}_F(t) \tag{4-3}$$

where

$$\underline{F}_F = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1/\tau_{DF} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1/\tau_{DF} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1/\tau_{AF} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1/\tau_{AF} \end{bmatrix}$$

with

$$\underline{G}_F = \begin{bmatrix} \underset{(4\text{x}4)}{\underline{0}} \\ \hline \underset{(4\text{x}4)}{\underline{I}} \end{bmatrix}$$

and

$$\underline{w}_F = [w_{DF1}(t) w_{DF2}(t) w_{AF1}(t) \; w_{AF2}(t)]^T \qquad (4\text{-}4)$$

such that

$\underline{x}_F(t)$ = filter state vector

$\underline{F}_F(t)$ = filter plant matrix

$\underline{w}_F(t)$ = zero-mean white Gaussian noise vector

$\underline{G}_F$ = noise input matrix

and

τ_{DF} = correlation time assumed for target acceleration

τ_{AF} = correlation time assumed for atmospheric jitter

The statistics of the noise processes are

$$E[\underline{w}_F(t)] = \underline{0}$$

$$E[\underline{w}_F(t)\underline{w}_F^T(t+\tau)] = \underline{Q}_F\ \delta(\tau)$$

where

$$\underline{Q}_F = \begin{bmatrix} 2\sigma_{DF}^2/\tau_{DF} & 0 & 0 & 0 \\ 0 & 2\sigma_{DF}^2/\tau_{DF} & 0 & 0 \\ 0 & 0 & 2\sigma_{AF}^2/\tau_{AF} & 0 \\ 0 & 0 & 0 & 2\sigma_{AF}^2/\tau_{AF} \end{bmatrix}$$

is the noise covariance kernel descriptor that yields the desired RMS noise values in which

σ_{DF}^2 = the assumed target acceleration process variance

σ_{AF}^2 = the assumed atmospheric jitter process variance.

4.3 State Propagation of the Gauss-Markov Filter Model

Since the state equations of the GM model are linear, they can be propagated in time using the conventional Kalman filter propagation equations. For propagation from one sample time t_i to the next t_{i+1} these equations are

$$\hat{\underline{x}}(t_{i+1}^-) = \underline{\Phi}_F(t_{i+1},t_i)\hat{\underline{x}}(t_i^+) \tag{4-6}$$

and

$$\underline{P}(t_{i+1}^+) = \underline{\Phi}_F(t_{i+1},t_i)\underline{P}(t_i^-)\underline{\Phi}_F^T(t_{i+1},t_i) + \underline{Q}_{FD}(t_{i+1},t_i) \tag{4-7}$$

where

$\underline{\Phi}_F(t_{i+1},t_i)$ = the filter state transition matrix

$\underline{P}(t_i^-)$ = the conditional state covariance matrix after the measurement update at time t_i

$\underline{P}(t_{i+1}^+)$ = the conditional state covariance matrix after propagation from time t_i to t_{i+1}

Also,

$$\underline{Q}_{FD}(t_{i+1},t_i) = \int_{t_i}^{t_{i+1}} \underline{\Phi}_F(t_{i+1},\tau)\underline{G}_F\underline{Q}_F\underline{G}_F^T\underline{\Phi}_F^T(t_{i+1},\tau)d\tau \tag{4-8}$$

which represents the growth in the uncertainty of the state estimate due to model uncertainty and the addition of noise between measurements. Since the filter plant matrix, E_F is constant, the state transition matirx, which must satisfy

$$\Phi_F(t,t_i) = E_F\Phi_F(t,t_i) \qquad (4-9)$$

and

$$\Phi_F(t_i,t_i) = I \qquad (4-10)$$

will be a function only of the sampling period ($\Delta t = t_{i+1}-t_i$) as opposed to being a function of t_{i+1} and t_i separately.

Therefore,

$$\Phi_F(t_{i+1},t_i) = \Phi_F(\Delta t) = \begin{bmatrix} 0 & 0 & \Delta t & 0 & J1 & 0 & 0 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & J1 & 0 & 0 \\ 0 & 0 & 1 & 0 & J2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & J2 & 0 & 0 \\ 0 & 0 & 0 & 0 & J3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & J3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & J4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & J4 \end{bmatrix} \qquad (4-11)$$

where

$$J1 = \tau_{DF}[\Delta t - \tau_{DF}(1 - e^{-\Delta t/\tau_{DF}}]$$

$$J2 = \tau_{DF}[1 - e^{-\Delta t/\tau_{DF}}]$$

$$J3 = e^{-\Delta t/\tau_{DF}}$$

$$J4 = e^{-\Delta t/\tau} AF$$

Similarly, Q_{FD} is also a function only on the sampling period. Evaluating the integral in Eq (4-8) yields a constant matrix of the form

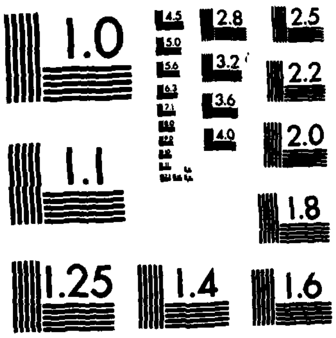
$$Q_{FD} = \begin{bmatrix} A1 & 0 & A2 & 0 & A3 & 0 & 0 & 0 \\ 0 & A4 & 0 & A5 & 0 & A6 & 0 & 0 \\ A2 & 0 & A7 & 0 & A8 & 0 & 0 & 0 \\ 0 & A5 & 0 & A9 & 0 & A10 & 0 & 0 \\ A3 & 0 & A8 & 0 & A11 & 0 & 0 & 0 \\ 0 & A6 & 0 & A10 & 0 & A12 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & A13 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & A14 \end{bmatrix} \qquad (4\text{-}12)$$

Results of the exact integration of non-zero terms of Eq. (4-8) can be found in subroutine FILTER in the Fortran source code in Appendix G.

4.4 Constant Turn-Rate Acceleration Model

In contrast to the first order GM acceleration model, the constant turn-rate (CTR) acceleration model considers the target dynamics, after being projected onto the FLIR image plane, to be well represented as constant speed, constant turn-rate manuevers. Although this model more accurately portrays typical airborne target dynamics than

AD-A124 781 ENHANCED IMAGE TRACKING: ANALYSIS OF TWO ACCELERATION 2/6
MODELS IN TRACKING..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. M R KOZEMCHAK
UNCLASSIFIED DEC 82 AFIT/GEO/EE/82D-4 F/G 17/7 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

the GM model (Ref: 3, 11, 19), it is implemented with the cost of additional filter computation due to the nonlinearity of the filter's description of jerk level motion (the time derivative of acceleration).

Specifically, the CTR acceleration dynamics model is (Refs: 3, 11, 19)

$$\dot{\underline{a}} = -\omega^2 \underline{v} + \underline{w}_{DF} \tag{4-13}$$

which implies a state differential equation

$$\dot{\underline{x}}_F(t) = \underline{f}[\underline{x}_F(t),t] + \underline{G}_F \underline{w}_F(t) = \begin{bmatrix} v_x \\ v_y \\ a_x \\ a_y \\ -\omega^2 v_x + w_{DF1}(t) \\ -\omega^2 v_y + w_{DF2}(t) \\ -(1/\tau_{AF})x_A + w_{AF1}(t) \\ -(1/\tau_{AF})x_A + w_{AF2}(t) \end{bmatrix} \tag{4-14}$$

such that,

$$\underline{x}_F(t) = [x_D, y_D, v_x, v_y, a_x, a_y, x_A, y_A]^T \tag{4-15}$$

which are the same eight states as defined by Eq (4-1); and

$$\omega = \frac{|\underline{v} \times \underline{a}|}{|\underline{v}|^2} = \frac{v_x a_y - v_y a_x}{v_x^2 + v_y^2} \tag{4-16}$$

where

$\underline{f}[\underline{x}_F(t),t]$ = a nonlinear function of the filter states

ω = the magnitude of the target's turn-rate on the FLIR image plane

$\underline{w}_F$ = zero mean, white, Gaussian driving noise vector

$\underline{G}_F$ = noise input matrix

τ_{AF} = correlation time assumed for the atmospheric jitter

The statistics of the noise vector are

$$E[\underline{w}_F(t)\underline{w}_F(t+\tau)] = \underline{Q}_F\delta(\tau) \qquad (4\text{-}17a)$$

$$E[\underline{w}_F(t)\underline{w}_F(t+\tau)] = \underline{Q}_F\,\delta(\tau) \qquad (4\text{-}17b)$$

where

$$\underline{Q}_F = \begin{bmatrix} \sigma_{DF}^2 & 0 & 0 & 0 \\ 0 & \sigma_{DF}^2 & 0 & 0 \\ 0 & 0 & \frac{2\sigma_{AF}^2}{\tau_{AF}} & 0 \\ 0 & 0 & 0 & \frac{2\sigma_{AF}^2}{\tau_{AF}} \end{bmatrix} \qquad (4\text{-}17)$$

is the covariance kernel descriptor that yields the desired RMS noise values when

τ_{DF}^2 = the assumed target acceleration white process power spectral density value

τ_{AF}^2 = the assumed atmospheric jitter Markov-1 process variance.

4.5 State Propagation of the Constant Turn-Rate Model

In terms of the state differential equation, Eq (4-2), and discrete time instants, the state estimate propagation can be written as

$$\hat{x}(t_{i+1}^-) = \hat{x}(t_i^+) + \int_{t_i}^{t_{i+1}} f[\hat{x}(t/t_i),t]dt \tag{4-18}$$

or equivalently

$$\hat{x}(t_{i+1}^-) = \hat{x}(t_i^+) + \int_{t_i}^{t_{i+1}} \dot{\hat{x}}(t/t_i),dt \tag{4-19}$$

Using a first order Euler integration approximation to Eq (4-19) over the interval $\Delta t = t_{i+1} - t_i$ yields

$$\hat{x}(t_{i+1}^-) = \hat{x}(t_i^+) + \dot{\hat{x}}(t_i^+)\Delta t$$

$$= \hat{x}(t_i^+) + f[\hat{x}(t_i^+),t_i]\Delta t \tag{4-20}$$

which is based on the most recent state estimate, $\hat{x}(t_i^+)$. Equation (4-20) neglects second and higher order terms by assuming that the state's time derivative is constant during each interval. This approximation is valid when t is sufficiently small when compared to the natural transient times of the physical system.

Similarly, the covariance propagation must also be written in terms of the most recent state estimates since

$$\underline{F}(t_i) = \left.\frac{\partial \underline{f}[\underline{x}(t/t),t]}{\partial \underline{x}}\right|_{\underline{x}=\hat{\underline{x}}(t^+)} \tag{4-21}$$

is no longer constant as it was in the GM filter. In other words, (Eq 4-21) assumes that the time varying $\underline{F}(t)$ can be approximated by the piecewise constant function $\underline{F}(t_i)$. This quasi-static assumption is valid, again, when Δt is small compared to the rate that $\underline{F}(t)$ varies.

Therefore, standard covariance propagation equations can be used with the restriction that

$$\underline{\Phi}_F(t_{i+1},t_i) = \exp(\underline{F}(t_i)\Delta t) \tag{4-22}$$

and thus

$$\underline{Q}_{FD}(t_{i+1},t_i) = \int_{t_i}^{t_{i+1}} \underline{\Phi}_F \underline{G}_F \underline{Q}_F \underline{G}_F^T \underline{\Phi}_F^T d\tau \tag{4-23}$$

be re-calculated each sample period. However, there is significant computational loading on the filter due to Eqs (4-22) and (4-23). To avoid much of the real time computation, the upper-left 6x6 portion of the state transition matrix in Eq (4-22) is truncated to first order terms, as in Eq (4-24).

$$\underline{\Phi}_F(t_{i+1},t_i) = \underline{I} + \underline{F}(t_i)\Delta t \tag{4-24}$$

This portion is instrumental in the covariance propagation of the states related to the target dynamics. The remaining portion, which is associated with the two atmospheric states, is time invariant and can be determined in its exact closed form. The derivation of $\underline{F}(t_i)$ and $\underline{\Phi}_F(t_{i+1},t_i)$ can be found in Appendix B. For the same reasons, the $\underline{Q}_{FD}$ matrix was also simplified. One consideration is simply to use the constant $\underline{Q}_{FD}$ matrix calculated for the Gauss-Markov model. Such an implementation is reasonable since realistic correlation coefficients are maintained in the off-diagonal terms. Another possibility is to approximate Eq (4-23) by

$$\underline{Q}_{FD} = \underline{G}_F \underline{Q}_F \underline{G}_F^T \Delta t \tag{4-25}$$

The latter is probably more appropriate for use in the CTR model since this model is a truer representation of the actual target dynamics; and consequently, introduces much smaller uncertainties into the state propagation. The form of the $\underline{Q}_{FD}$ matrix given by Eq (4-25) is such that all elements are zero except for the four lower-right diagonal elements.

4.6 Measurement Update Equations

It is because of the nonlinear characteristics of the FLIR measurements that a simple linear Kalman filter cannot be used alone in the HEL pointing and tracking problem. Specifically, this is so because the intensity function acquired from FLIR measurements is a nonlinear function of the system states (for both the Gauss-Markov and Constant Turn-Rate models). In general, the measurement equation has the form

$$z_{kl}(t_i) = h_{kl}[\underline{x}(t_i),t_i] + v_{kl}(t_i) \tag{4-26}$$

which is equivalent to Eq (3-53) expressed in terms of the kl-th pixel. After propagating the state estimate from time t_{i-1} to t_i, it is desired that the filter use the estimate along with the 64 pieces of new measurement information represented in Eq (4-26) to determine an updated estimate of the target's centroid location on the image plane. To process this informatin, an extended Kalman filter is thought to be best suited for the task since it accounts for nonlinearities in the measurement equation (to a degree, and this is superior to a linearized Kalman filter) and is not as burdensome as higher order nonlinear filters (Ref: 8, Ch 11,12). In effect, the extended Kalman filter relinearizes the measurement equation every sample period about the most recent estimate of the system's state, $\hat{\underline{x}}(t_i)$.

Since computation time is critical in this application, the inverse covariance form of the measurement update equations is beneficial since it does not require the on-line inversion of a 64x64 matrix (since $\underline{K} = \underline{P}^{-}\underline{H}^T(\underline{H}\underline{P}^{-}\underline{H}^T+\underline{R})^{-1}$) at every update. In contrast, when the update equations are formulated in this manner only two 8x8 inversions are required on-line. Thus, the equations can be written as:

$$\underline{P}^{-1}(t_i^+) = \underline{P}^{-1}(t_i^-) + \underline{H}^T(t_i)\underline{R}^{-1}(t_i)\underline{H}(t_i) \tag{4-27}$$

$$\underline{P}^{-1}(t_i^+) = [\underline{P}^{-1}(t_i^+)]^{-1} \tag{4-28}$$

$$\underline{K}(t_i) = \underline{P}(t_i^+)\underline{H}^T(t_i)\underline{R}^{-1}(t_i) \qquad (4\text{-}29)$$

$$\hat{\underline{x}}(t_i^+) = \hat{\underline{x}}(t_i^-) + \underline{K}(t_i)[\underline{z}(t_i) - \underline{h}(\hat{\underline{x}}(t_i^-),t_i)] \qquad (4\text{-}30)$$

where

$\underline{h}[\hat{\underline{x}}(t_i^-),t_i]$ = a nonlinear intensity function written in terms of the most recent state estimate, $\underline{x}(t_i^-)$

$$\underline{H}(t_i) = \left.\frac{\partial \underline{h}[\underline{x}(t_i),t_i]}{\partial \underline{x}}\right|_{\underline{x}(t_i) = \hat{\underline{x}}(t_i^-)}$$

= the linearized function of the intensity measurements evaluated at the most recent state estimate

$\underline{P}(t_i^-)$ = the conditional covariance matrix after propagation to time t_i and before the measurement update at that time

$\underline{P}(t_i^+)$ = the conditional covariance matrix after the measurement update at time t_i

$\underline{K}(t_i)$ = Kalman filter gain

$\underline{z}(t_i)$ = the actual realization of the measurement vector at time t_i.

The method used to derive the nonlinear and linearized intensity functions involves the use of the FFT, phase shifting and smoothing techniques as presented in Chapter II.

4.7 Dynamic Driving Noise Estimation

The dynamic driving noise matrix, Q_{FD}, is used to model the statistics expected of the target's dynamics. Since the targets being tracked may exhibit a variety of dynamic characteristics, the "optimal" dynamic driving noise in the Kalman filter model will not be constant. If variances are chosen large enough, use of a constant dynamic driving noise matrix (Q_{FD}) may enable the filter to maintain track on maneuvering targets; however, less than optimal performance is obtained when the target exhibits a benign trajectory.

In this research, a time varying dynamic driving noise is appropriate in two instances: when the filter is in an acquisition mode, and when maximum tracking efficiency is desired in regard to the relationship between actual tracking errors and the filter's estimate of its own errors. During acquisition, the filter is handed information about the target's position and velocity from some device such as radar. To deal with inaccuracies in this information, the filter's initial state covariance matrix, P_0, and driving nise, Q_{FD}, are raised to values above what they would normally be during steady state operation. The Q_{FD} matrix is then reduced linearly until a reasonable level is reached or the adaptive mode is initiated. Meanwhile, the state covariance matrix is allowed to propagate and update as usual.

Methods available for Q_{FD} adaptation include maximum likelihood techniques, Bayesian and multiple model algorithms, as well as correlation and covariance matching techniques (Ref: 8, Ch 10). In this research an approximation to the maximum likelihood technique is investigated rather than a full scale likelihood estimator since the latter involves significant computational loading. Also, a crude simulation of multiple modelling can be implemented by giving the filter knowledge of the time a maneuver is initiated and artificially increasing an otherwise time-invariant Q_{FD} matrix.

Specifically, the Q_{FD} estimator can be derived as follows. Given the covariance propagation equation,

$$\underline{P}(t_i^-) = \underline{\Phi}_F(t_i,t_{i-1})\underline{P}(t_{i-1}^+)\underline{\Phi}_F^T(t_i,t_{i-1}) + \underline{Q}_{FD}(t_{i-1}) \quad (4\text{-}31)$$

and,

$$\underline{P}(t_i^+) = \underline{P}(t_i^-) - \underline{K}(t_i)\underline{H}(t_i)\underline{P}(t_i^-) \quad (4\text{-}32)$$

solving for Q_{FD} while keeping the latter term in Eq (4-32) intact yields:

$$\underline{Q}_{FD}(t_{i-1}) = \underline{K}(t_i)\underline{H}(t_i)\underline{P}(t_i^-) + \underline{P}(t_i^+) - \underline{\Phi}_F(t_i,t_{i-1})\underline{P}(t_{i-1}^+)\underline{\Phi}_F^T(t_i,t_{i-1}) \quad (4\text{-}33)$$

All of the quantities on the right hand side of the above equation are available except the first term. This term will be estimated from the filter residuals. Rewriting the measurement update equation, the state update can be expressed as

$$\hat{\underline{x}}(t_i^+) - \underline{x}(t_i^-) = \Delta\underline{x}(t_i) = \underline{K}(t_i)\underline{r}(t_i) \qquad (4\text{-}34)$$

The sequence, $\Delta\underline{x}(t_i)$, is a white sequence with covariance $\underline{K}(t_i)\underline{H}(t_i)\underline{P}(t_i)$ and is independent of $\underline{x}(t_i)$ (Ref: 8, Ch 10).

$$E[\Delta\underline{x}(t_i)\,\Delta x(t_i)^T] = \underline{K}(t_i)\underline{H}(t_i)\underline{P}(t_i^-) \qquad (4\text{-}35)$$

Then, assuming the filter is essentially in steady state, an ergodic approximation over the N most recent samples yields:

$$E[\Delta\underline{x}(t_i)\Delta\underline{x}(t_i)^T] = \frac{1}{N}\sum_{j=i-N+1}^{i} [\Delta\underline{x}(t_j)\,\Delta\underline{x}(t_j)^T] \qquad (4\text{-}36)$$

Using Eqs (4-33), (4-35), and (4-36), an estimate of $\underline{Q}_{FD}$ can be made.

$$\hat{\underline{Q}}_{FD}(t_i) = \frac{1}{N}\sum_{j=i-N+1}^{i} [\Delta\underline{x}(t_j)\Delta\underline{x}(t_j)^T] + \underline{P}(t_i^+)$$

$$- \underline{\Phi}_F(t_i,t_{i-1})\underline{P}(t_{i-1})\underline{\Phi}_F^T(t_i,t_{i-1}) \qquad (4\text{-}37)$$

Including the latter two terms within the time average gives the result

$$\hat{\underline{Q}}_{FD}(t_i) = \frac{1}{N}\sum_{j=i-N+1}^{i} \{[\Delta\underline{x}(t_j)\Delta\underline{x}(t_j)^T] + \underline{P}(t_j^+)$$

$$- \underline{\Phi}_F(t_j,t_{j-1})\underline{P}(t_{j-1})\underline{\Phi}_F^T(t_j,t_{j-1})\} \qquad (4\text{-}38)$$

all of whose terms are already calculated during normal filter updates and propagations. This finite memory result can be further simplified through a fading memory approximation in order to reduce storage requirements. Specifically,

$$\hat{\underline{Q}}_{FD}(t_i) = k\hat{\underline{Q}}_{FD}(t_{i-1}) + (1-k)\ \underline{Q}_{FD1}(t_i) \qquad (4\text{-}39)$$

where $\underline{Q}_{FD1}(t_i)$ is single term within the summation in Eq (4-38) when j=i. The parameter, k, is what determines the responsiveness of this estimate; it is required to be between 0 and 1. A value close to 1 will result in very little weight put on the new estimates yielding a very slowly responding function. In contrast, a value close to 0 will cause the estimate to respond rapidly. As the value of k decreases; however, the validity of the variance approximation of the residuals becomes questionable since a shorter time average is being employed. Typically, the value of k is 0.8.

V. Performance Analysis Methodology

5.1 Introduction

This chapter presents the methodology used to test the performance of the extended Kalman filter tracking algorithms against the truth model used to represent the real world. Since the real world constantly changes, causing no two experiments ever to be exactly the same, a Monte Carlo analysis is used as the method of evaluation. This method is presented in the first section of this chapter. Following it are discussions of the actual figures of merit that can be obtained from the Monte Carlo analysis and a description of the parameters varied in the computer simulations.

5.2 Monte Carlo Analysis

In essence, a Monte Carlo analysis involves running numerous sample-by-sample simulations of a single truth model trajectory that differs from run to run only by the effects of the noise sources within the model. In each simulation, error statistics are computed between desired quantities of the truth model states and the extended Kalman filter states. Ideally, an infinite number of simulations should be performed in order to ensure that sample statistics are a true representation of the actual error characteristics.

However, to be realistic, this number has to be finite due to limited computer resources available. It has been shown (Ref: 5; 12, p 42) that 20 simulations is sufficient to indicate the convergence of the error statistics of interest for this problem context. In this research, the number of simulations had to be limited even further, to one half of this value, due to extensive time requirements of the computer simulation. The same studies indicated a reasonable convergence of the error statistics at this number of simulations as well. In any event, the errors committed at any one sample time in any one simulation can be determined as indicated by Figure 15 (Ref: 7, p 327; 16, p 169).

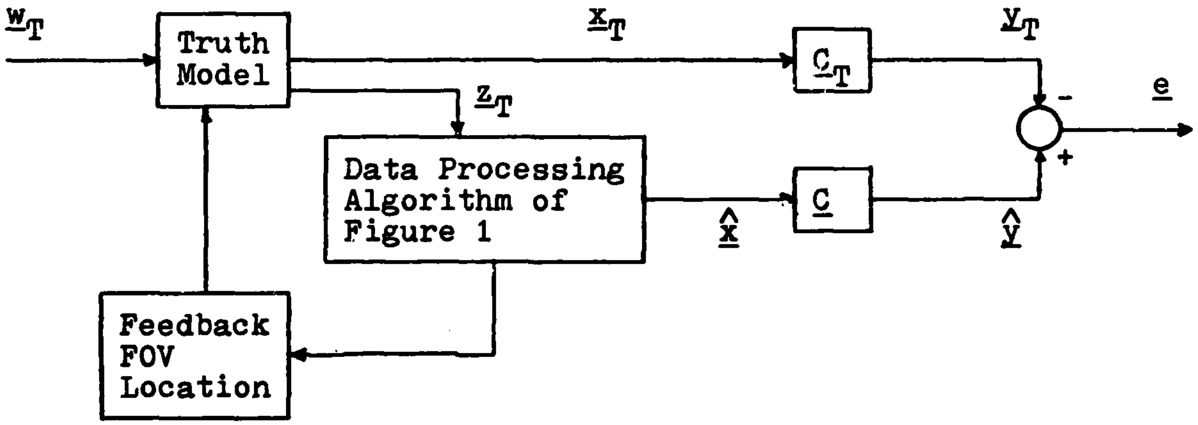


Figure 15. Generation of Error Statistics

The truth model generates the measurement process, $\underline{z}_T(\cdot,\cdot)$, from the driving white Gaussian noise and from information fed back from the filter on the location of the field of view. The data processing algorithm uses realizations of the discrete time process, $\underline{z}_T(t_i,w_k) = \underline{z}_T(t_i)$, at each time t_i to generate the state estimate process, $\hat{\underline{x}}(\cdot,\cdot)$. In order to related the critical quantities of the truth model states to the filter state estimates the following linear time-invariant transformations are made:

$$\hat{\underline{y}}(t_i) = \underline{C}\hat{\underline{x}}(t_i) \tag{5-1}$$

and

$$\underline{y}_T(t_i) = \underline{C}_T\underline{x}_T(t_i) \tag{5-2}$$

where $\underline{C}$ and $\underline{C}_T$ are selection matrices used to choose the appropriate states for comparison purposes. These results can then be combined to determine the filter errors before and after measurement incorporation at a time t_i by

$$\underline{e}_T(t_i^-,\cdot) = \hat{\underline{y}}(t_i^-,\cdot) - \underline{y}_T(t_i,\cdot) \tag{5-3}$$

and

$$\underline{e}_T(t_i^+,\cdot) = \hat{\underline{y}}(t_i^+,\cdot) - \underline{y}_T(t_i,\cdot) \tag{5-4}$$

5.3 Figures of Merit

The quantities that were used in this research to characterize the filter's tracking performance were the errors in the estimated values of $x_D(t_i-)$, $y_D(t_i+)$, $x_D(t_i-)$, $y_D(t_i+)$, $x_{CEN}(t_i+)$, $y_{CEN}(t_i+)$, $v_x(t_i+)$, $v_y(t_i+)$, $a_x(t_i+)$, and $a_y(t_i+)$. Probably the most important estimate, that of the dynamics position, was evaluated both before and after measurement updates (in most cases) in order to determine how well the internal dynamics model and the measurement data processing algorithm performed. The remaining estimates were only evaluated after measurement incorporation to limit the amount of analysis required for each Monte Carlo study. Statistics calculated in the error processing routines of the computer code include the ensemble error means and the error variances of these variables at each time frame of the simulations. An example of a calculation of the mean error is

$$\bar{E}_{x_{D_i}} = \frac{1}{N}\sum_{k=1}^{N}\left(x_{T_{D_{ik}}} - \hat{x}_{D^-_{ik}}\right) = \frac{1}{N}\sum_{k=1}^{N} e_{x_{D_{ik}}} \qquad (5\text{-}5)$$

where

$\bar{E}_{x_{D_i}}$ = mean error (ensemble average over all simulations) of x dynamics prior to the measurement update at time t_i

$x_{T_{D_{ik}}}$ = value of the truth model x dynamics state for frame i and simulation k

$\hat{x}_{D^-_{ik}}$ = value of the filter estimates x dynamics state for simulation k prior to the measurement update at time t_i

N = the number of simulations performed

with the variance of the error given by

$$\sigma^2_{e_{x_{D_i}}} = \frac{1}{N-1} \sum_{k=1}^{N} (e_{x_{D_{ik}}} - \bar{E}_{x_{D_i}})^2 = \frac{1}{N-1} \sum_{k=1}^{N} e^2_{x_{D_{ik}}} - \frac{N}{N-1} \bar{E}^2_{x_{D_i}} \tag{5-6}$$

Similar equations can be written for the remaining quantities of interest. The results of equations such as Eqs (5-5) and (5-6) were then used by the plotting routine to generate two different types of plots. The first type displays the mean error along with the mean error plus/minus the error standard deviation at each sample time within the five second simulation. The second plot used to analyze filter performance compares the filter's estimate of the standard deviation of its own errors ($P_{jj}(t_i)$) to a quantity, defined here as the "true" root-mean-square (RMS) error, that reflects the magnitudes of both the true mean error and the true standard deviation. The RMS error is expressed as

$$E_{RMS_{ji}} = \sqrt{\bar{E}^2_{ji} + \sigma^2_{e_{ji}}} \tag{5-7}$$

where E and σ are as defined by Eqs (5-5) and (5-6) except that the subscript j now represents the particular quantity of interest. Samples of the plots are shown in Figures 16 and 17.

The data was further processed by the plot routine to determine the time-averaged mean error along with the time averaged error variance over the portions of the simulations

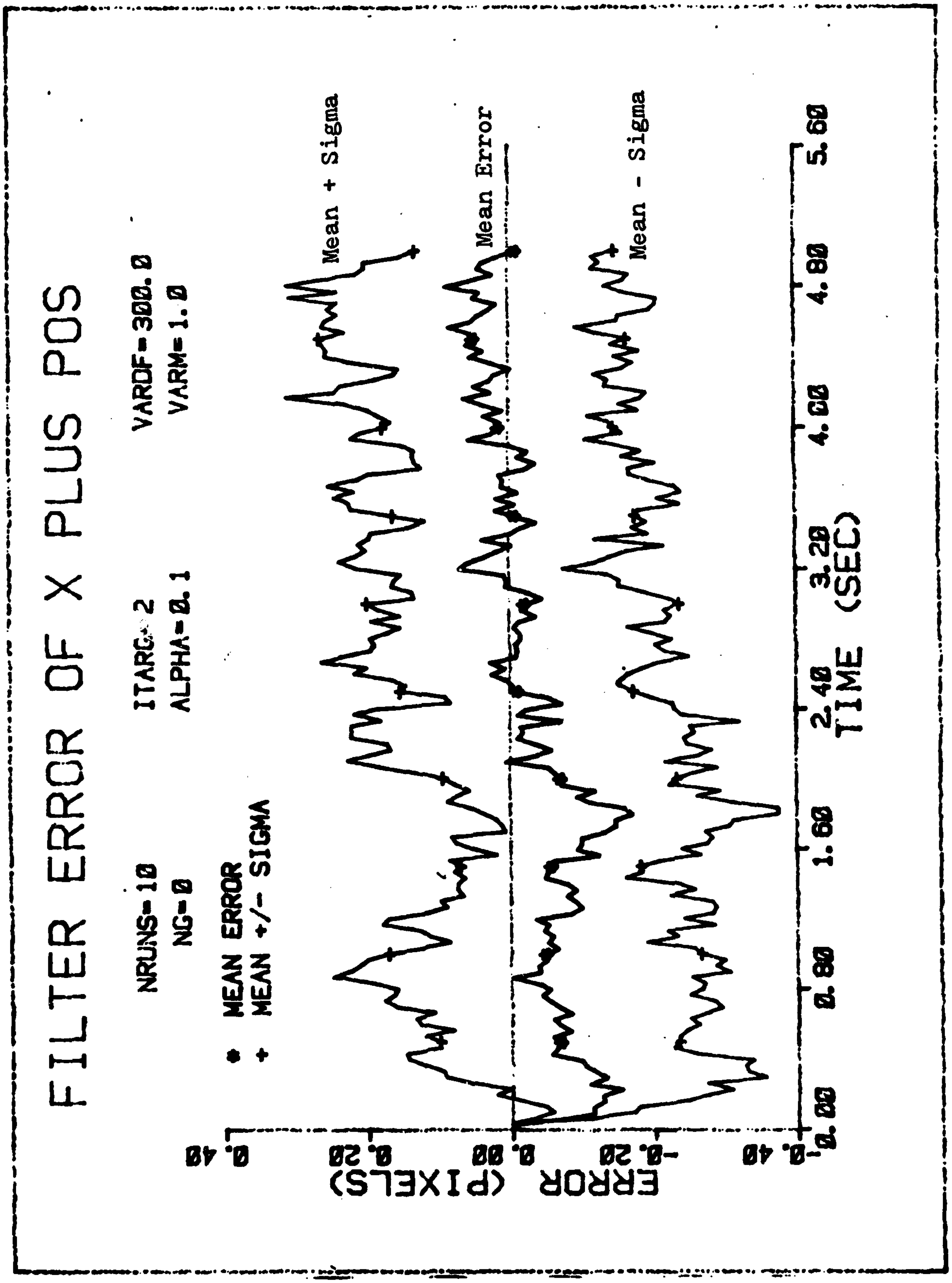


Figure 16. Sample Plot - Mean Error +/- Standard Deviation

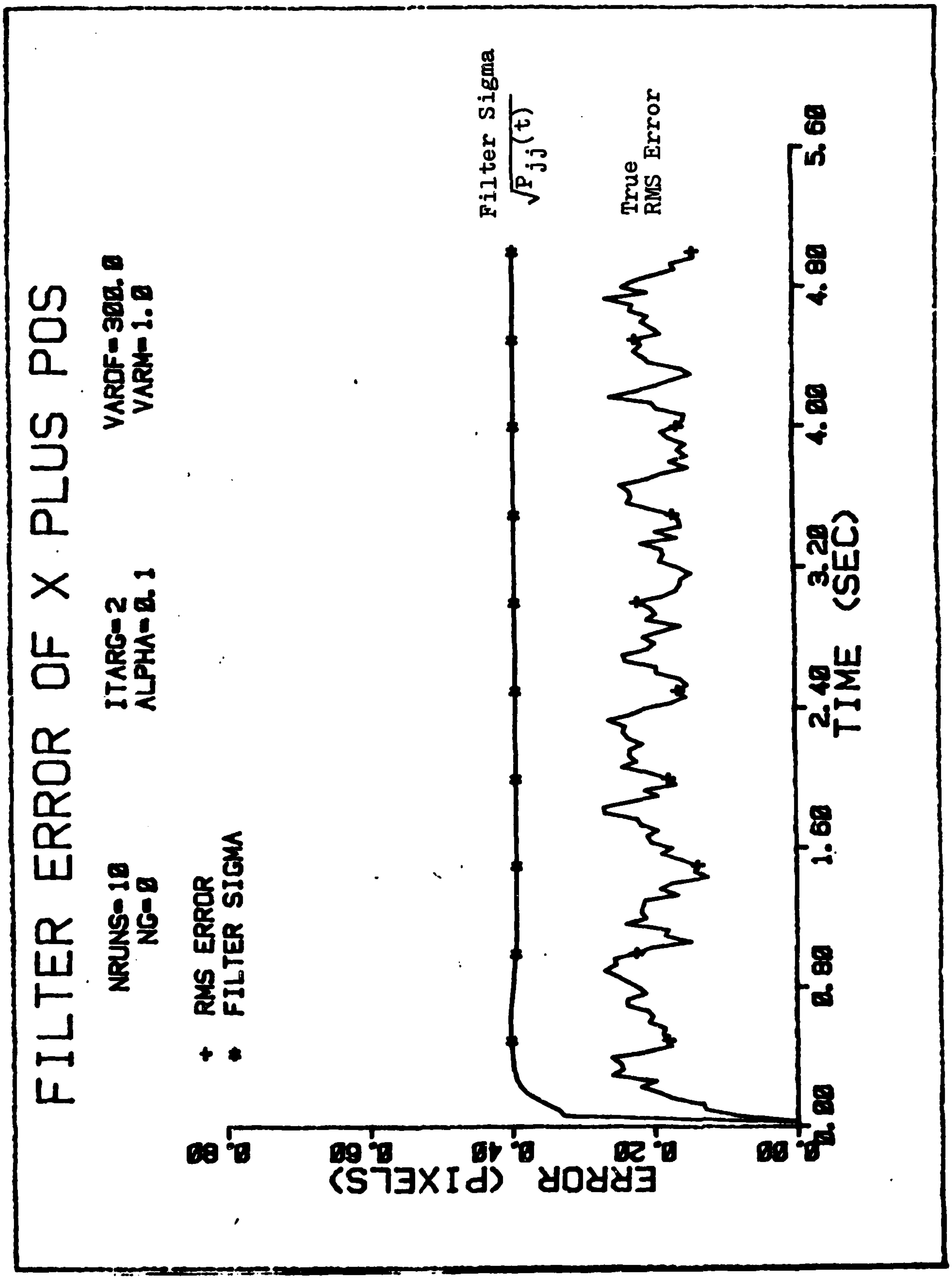


Figure 17. Sample Plot - Filter Sigma vs True RMS Error

in which transient effects due to acquisition and/or a change in target dynamics had diminished. Also determined was the peak error over the entire engagement. These results are displayed in the tables presented in the next chapter.

5.4 Variation of Parameters

In order to allow flexibility in testing the extended Kalman filter tracking algorithms, the computer simulation is set up to allow variation of many of the parameters that describe the filter design characteristics as well as the tracking environment. In the actual computer simulation, parameters are controlled by choosing the appropriate input data at the end of the computer code shown in Appendix G. Following is a description of these parameters and the different values implemented. Tables that relate sets of parameter values to corresponding performance plots are Table D-1 and Table E-1 for the GM and CTR filters respectively.

One of the most critical factors used to describe the tracking environment is the trajectory being simulated. This is partially controlled by the initial position, velocity, and acceleration values of the target dynamics specified in the inertial reference frame. The initial target position for all simulations was chosen to be $(x_0, y_0, z_0) = (5000, 500, 20000)$ meters. From this point in inertial space, a constant velocity trajectory was simulated until, in some cases, a constant G pull-up or a constant G turn was initiated. Specifically, four different trajectories were simulated as indicated in Table 5.1.

Table 5.1 Target Trajectories Simulated

Case	Initial Position	Initial Velocity	Initial Acceleration	Maneuver

1	x = 5000
	y = 500
	z = 20000
		$v_x = -1000$
		$v_y = 0$
		$v_z = 0$
			$a_x = 0$
			$a_y = 0$
			$a_z = 0$
				constant velocity trajectory

2	x = 5000
	y = 500
	z = 20000
		$v_x = -1000$
		$v_y = 0$
		$v_z = 0$
			$a_x = 0$
			$a_y = 0$
			$a_z = 0$
				constant G-pull-up in j direction at t = 2.0 seconds

3	x = 5000
	y = 500
	z = 20000
		$v_x = -700$
		$v_y = -300$
		$v_z = 500$
			$a_x = 0$
			$a_y = 0$
			$a_z = 0$
				constant velocity trajectory

4	x = 5000
	y = 500
	z = 20000
		$v_x = -1000$
		$v_y = 0$
		$v_z = 0$
			$a_x = 0$
			$a_y = 0$
			$a_z = 0$
				constant G turn in k direction at t = 2.0 seconds

The first three cases in Table 5.1 were discussed in greater detail in Chapter III, while the fourth case is identical to the second except for the direction of the constant-G maneuver. The first and third trajectories are relatively benign and are used to determine basic filter performance.

The remaining two involve significant changes in target dynamics. To a large degree, the dynamics of Trajectory 2 remain in the plane perpendicular to the line of sight. Trajectory 4 involves a maneuver away from this plane. Also, compared to that of Trajectory 2, the tracker is exposed to a different target image in Trajectory 4. Particular maneuvers are further described by specifying the number of Gs, the roll-rate, as well as the maneuver initiation times. Values of 0, 2, and 5 were used in the constant-G maneuvers while 0.0, 0.5, 1.0, and 2.0 were used as constant roll-rates. An important parameter used in the computer code to describe the target is the target selection flag. This parameter determines whether the target to be tracked will produce a single or multiple hot-spot image with a static or dynamic image profile. Setting this flag equal to 0 will produce a multiple hot-spot image with three hot-spots (each individually displaying circular equal-intensity contours) fixed in relation to one another on the image plane as shown in Figure 18. In contrast, a value of 1 again produces a multiple hot-spot image (see Figure 19); however, now the three spots will move about with respect to one another and change sizes and shapes according to realistic projections of 3-dimensional targets onto the FLIR image plane. In this case, provisions were also made for portions of the target, and the corresponding hot-spots, to be obscured from the view of the sensor. Finally, when the target flag equals 2 a single hot-spot is produced with

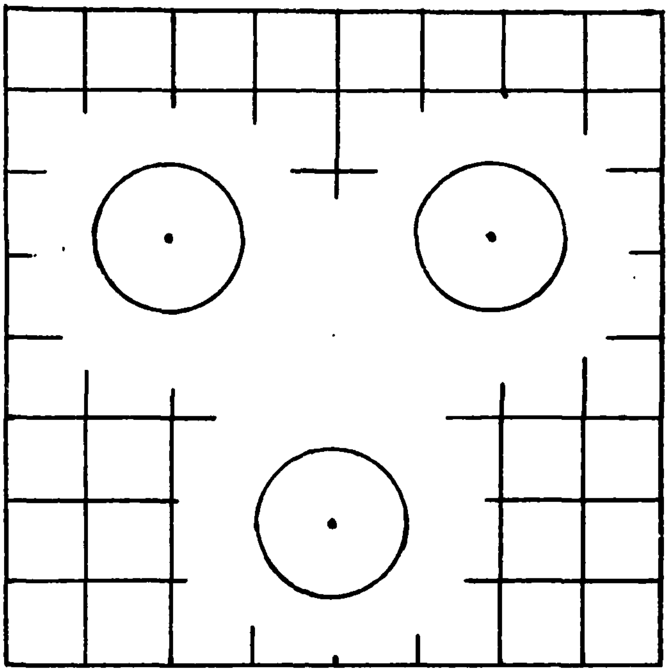


Figure 18. Static Multiple Hot-Spot Target Image

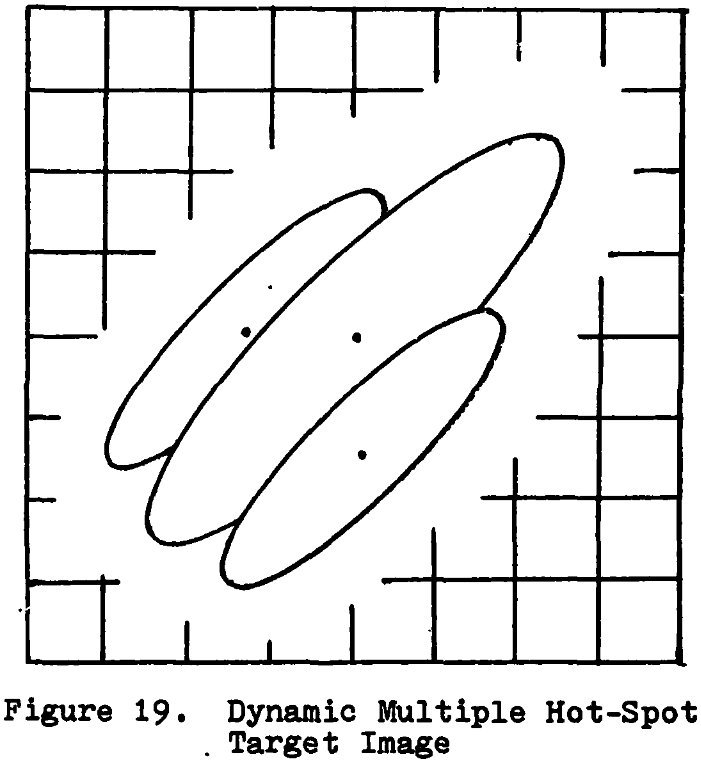


Figure 19. Dynamic Multiple Hot-Spot Target Image

circular equal-intensity contours. When this flag is either 0 or 2, the individual Gaussian intensity distributions are described by a glint dispersion parameter, σ_g^2, of either 2.0 or 3.0. In contrast, when the flag is 1, a more detailed description of the target image is possible. Four arrays (σ_{vo}, σ_{pvo}, δ_v, and δ_{pv}), each containing three elements apiece (units of pixels), are used to specify the size, shape and relative orientation of the three corresponding bivariate Gaussian intensity hot-spots at a reference range of 20000 meters. These values specify the maximum possible image dimensions when the target is located at the indicated reference range. The values were set to constants for all simulations. Specifically, $\delta_v = (0.64, -0.86, -0.86)$ represents the maximum pixel distance from the centroid location to the peaks of the three hot-spots on the FLIR image plane in the direction of the image's velocity vector. Similarly, $\delta_{pv} = (0.0, 2.5, -2.5)$ describes equivalent distances in the direction perpendicular to the velocity vector. In addition, the array $\sigma_{vo} = (4.0, 1.5, 1.5)$ contains the possible dispersions of the three Gaussian intensity distributions along the velocity direction of the target image. Similarly, $\delta_{pvo} = (1.0, 1.0, 1.0)$ describes the image dispersions in the direction perpendicular to the velocity. Figures 10 and 11 describe the relationship of these parameters.

Other parameters that further characterize the tracking environment are the maximum signal intensity, the variance of the spatially correlated background/FLIR noise, and the standard deviation of the truth model atmospheric jitter. In essence the parameter used to represent the total noise corruption of the measurement data is chosen to reflect both FLIR and background noises; however, it should be noted that true FLIR noise is not spatially correlated. The truth model background noise variance along with the maximum signal intensity, determine the signal to noise ratio (SNR) of the target image. Specifically,

$$\frac{S}{N} = \frac{\text{Imax - mean background noise}}{\text{RMS value of background noise}} = \frac{\text{Imax}}{\sigma_{b2}} \qquad (5\text{-}8)$$

when the noise is zero mean. The superior performance of Kalman filter trackers over standard correlator trackers in low signal to noise environments has been demonstrated repeatedly (Ref: 5, 12). Therefore, the parameters in Eq (5-8) were chosen in order to produce a signal to noise ratio of 20 in all simulations which is a reasonable value in realistic scenarios. Additional parameters held constant throughout most of the research included the standard deviation of the true atmospheric jitter at 0.1414 and the variance of the filter atmospherics at 0.2. It was desired that these two parameters model equivalent atmospheric effects; however, an oversight early in the research resulted on the use of these values.

In addition to the variance of the filter atmospherics, other parameters that specify filter design characteristics include the variance of the filter dynamics driving noise, the smoothing constant used in exponential smoothing, the number of zeros padded, and the number of high frequency components zeroed in the data processing algorithm used to generate the target reference image. The strength of the filter dynamics driving noise, σ_D^2, reflects the level of the target maneuvering ability and the uncertainty in the filter dynamics model. It is used in order to tune the filter for optimal tracking performance. The weighting factor of the exponential smoothing algorithm, α , is discussed in Section 2.3 Exponential Smoothing. This parameter is of particular interest when image variations due to changing size and shape become significant. Values used in this research included 0.1, 0.2, and 0.5.

The remaining parameters that specify filter design characteristics affect the measurement processing portion of the algorithm. It was indicated in Chapter II than an array of size 24x24 is processed, as opposed to the 8x8 FOV, in order to reduce problems associated with taking FFTs of finite size data arrays. A common practice would be simply to surround the 8x8 FOV array with eight rows and columns of zeros (i.e., a value of 8 for the zero padding parameter); however, this practice induces artificial high frequencies and edge effects especially when the actual target intensity function is not practically zero at the FOV boundaries. Such problems are reduced if the FOV is padded with an equivalent amount of measurement data (i.e., a value of 0 for the zero padding parameter). Such a luxury exists in this application due to the large size of the actual FLIR measurement array available. An alternative would be to pad with zeros and then perform high frequency spatial frequency filtering. This is done by specifying the number of high frequencies to be set to zero within the Fourier Transform of the image that is performed during the derivation of the intensity profile. It should be noted that spatial frequency fitlering may enhance or corrupt tracking performance, depending upon the measurement noise and the characteristics of the true target profile. Specifically, if the target image is smooth and shallow relative to the edge and noise effects, high frequency filtering will enhance tracking performance. In contrast, if the image displays sharp intensity variations filtering will produce errors in the resulting reference image and likely degrade tracking performance. Thus, the number of frequencies zeroed and number of zeros padded parameters were set to 0 for all simulations so as not to mask the effects of other parameters on filter performance.

The final parameters specified in each computer simulation are the number of frames per simulation and the number of runs in the Monte Carlo analysis. A value of 150 was used as the number of frames in order to produce a five second simulation of measurements being received at a 30 Hz rate. Finally, the number of runs per Monte Carlo simulation was limited to 10 due to the large amount of CPU time required per simulation.

5.5 Summary

In review, this chapter presented the methodology used to conduct a performance analysis of the extended Kalman filter tracking algorithms. The specific means of evaluation is a Monte Carlo analysis which computes the statistics of the filter estimates both before and after measurement incorporation at each time frame. The figures of merit determined at each time frame for a quantity of interest include the mean error and variance of that error, the filter's estimate of its own error variances, and the true RMS error. After transients have died out, time averaged mean and standard deviations are determined to give quick numerical insights into the overall performance of the filter. Peak errors are also determined over the same interval. The final portion of this chapter describes the input parameters used to specify the tracking environment and filter design charactersitics. The range of values used for these parameters are also listed there.

VI. Results

6.1 Introduction

This chapter presents the results of the performance analysis conducted on both the first order Gauss-Markov (GM) acceleration and constant turn-rate (CTR) acceleration extended Kalman filters. Also presented is a review of the performance of the adaptive Brownian Motion (BM) acceleration extended Kalman filter (using an $\underline{h}$ derived on the basis of a bivariate Gaussian intensity function) previously developed by Captains Harnly and Jensen (Ref: 5). Of particular interest, is the performance of this particular filter in the presence of a multiple hot-spot image. In Section 6.2, the importance of modelling the spatial correlations in the measurement noise is examined by comparing performance of the GM filter with the spatial correlations modelled to that without the correlations modelled. Section 6.3 then contrasts the BM, GM and CTR filters in the case of relatively benign trajectories, with targets displaying single hot-spot images as well as multiple hot-spot images. Then the performances of the three filters are examined when the target undergoes dynamic maneuvers such as constant G turns and constant roll-rate maneuvers. The cases displaying the GM and CTR filters also involve the dynamic image profile developed in this research. Finally, in Section 6.5 the efforts to implement the estimation of the dynamic driving noise matrix are discussed. The performance plots associated with the Gauss-Markov filter are presented in Appendix D with Table D.1 displaying the parameters of the various cases at the beginning of the appendix. Similarly, the plots of the constant-turn-rate filter appear in Appendix E, and the plots of Harnly and Jensen's filter in Appendix F. Unless otherwise specified, the parameter values are the standard parameters as listed in the previous chapter. Tables 6.1 and 6.2 present the figures of merit of all the cases involving the Gauss-Markov and constant-turn-rate filters in regard to the dynamics and centroid position tracking errors. The averaging done to obtain the figures in these tables was performed after transients had died out. In the case of constant velocity trajectories, the averaged interval began 0.5 seconds after simulation initiation. For cases involving constant G maneuvers initialized two seconds after simulation initiation, the interval over which averaging was performed was the last two seconds of the simulation (i.e., starting one second after maneuver initiation). Parameters pertaining to these cases are specified in Tables D.1 (for cases 1 through 11) and E.1 (for cases 12 through 24).

6.2 Modelling Spatial Correlations

As discussed in Chapter 3, the combined effects of FLIR and background noises were modelled as spatially correlated, zero mean, white, Gaussian noise of strength σ_b^2. In an

Table 6.1 Time Averaged Figures of Merit - Dynamics Position Estimates

	x Channel Dynamics			y Channel Dynamics		
Case	Peak Error	Mean Error	Stand. Dev.	Peak Error	Mean Error	Stand. Dev.
1	-.339	-.173	.194	.163	.084	.227
2	-.366	-.187	.391	-.205	-.068	.377
3	.291	.149	.509	-.404	-.249	.429
4	.312	.162	.509	-.411	-.254	.429
5	-.462	-.291	.466	.120	.041	.609
6	.272	.027	.160	-.212	-.039	.126
7	-.233	-.042	.190	-.148	-.024	.117
8	-.233	-.002	.189	-.173	-.048	.115
9	.378	.184	.148	-.379	-.247	.163
10	.310	.174	.132	-.114	.005	.123
11	-.239	-.070	.125	-.141	-.004	.132
12	.149	.090	.199	-.175	-.042	.229
13	-.229	-.106	.266	.137	.056	.372
14	-.169	-.029	.184	.108	.038	.219
15	.440	.369	.156	-.075	-.007	.093
16	.403	.325	.159	-.079	.002	.094
17	.313	.226	.139	-.604	-.495	.376
18	.435	.334	.168	.183	.063	.133
19	.274	.207	.136	-.127	-.016	.124
20	.402	.239	.167	-.186	-.071	.117
21	.539	.341	.151	-.284	-.100	.163
22	.644	.377	.190	-.240	-.091	.136
23	-.138	-.088	.213	-.125	-.038	.251
24	.513	.419	.161	-.163	-.048	.138

Table 6.2 Time Averaged Figures of Merit - Centroid Position Estimates

	x Channel Centroid			y Channel Centroid		
Case	Peak Error	Mean Error	Stand. Dev.	Peak Error	Mean Error	Stand. Dev.
1	-.211	-.148	.171	.126	.076	.199
2	-.220	-.161	.373	-.138	-.076	.353
3	-.099	-.026	.495	-.262	-.181	.410
4	-.100	-.023	.495	-.265	-.183	.410
5	-.296	-.266	.445	.092	.033	.597
6	.202	.058	.143	-.149	-.048	.078
7	-.121	-.016	.166	-.073	-.032	.044
8	.133	.024	.163	-.091	-.056	.048
9	.310	.152	.136	-.138	-.125	.075
10	.074	.026	.075	-.033	-.015	.049
11	-.107	-.045	.081	-.076	-.012	.084
12	.141	.073	.176	-.091	-.034	.205
13	-.191	-.158	.242	.112	.056	.362
14	-.148	-.079	.161	-.090	.038	.193
15	.401	.313	.145	-.046	-.011	.054
16	.366	.280	.145	.045	.001	.051
17	.376	.297	.160	-.563	-.531	.391
18	.391	.291	.142	.136	.065	.049
19	.102	.070	.076	-.035	-.014	.047
20	.285	.190	.143	-.175	-.072	.061
21	.483	.291	.125	-.186	-.101	.128
22	.522	.327	.166	-.219	-.092	.088
23	-.169	-.144	.213	-.088	-.051	.249
24	.425	.325	.127	-.098	-.063	.082

effort to determine the sensitivity of the filter's performance to its model of such noise, runs were made in which the measurement noise was portrayed by the filter to be spatially correlated as well as in which it was modelled as spatially uncorrelated noise. In both cases, the strength of the filter-assumed noise was chosen as twice the value of the actual (truth model) noise strength in order to reflect uncertainty in the internal model, providing reasonably good filter tuning. The two cases of interest were plotted for the filter with the Gauss-Markov acceleration model (cases 1 and 2 of Table D.1) in which a target with a single spot image travelled along Trajectory 1 of Table 5.1. The plots corresponding to spatially correlated noise (case 1) are shown in Figures D.1 through D.14 while those with spatially uncorrelated noise (case 2) are shown in Figures D.15 through D.20. Comparison of these plots indicates the significance of correctly modelling the spatial correlations. Examining the peak and time averaged mean errors of the position estimates of both cases (Table 6.1) indicates only slight increase in the magnitudes of the errors in either x or y channel of less than 0.05 pixels (although the y channel does change sign). However, there is substantial degradation of the time averaged standard deviation of the errors when spatial correlations are unmodelled. The standard deviation increases by more than 100%. (0.2 pixels) in the x channel and 65% (0.15 pixels) in the y channel. Although they were not plotted in this case, such increases make the RMS errors of the position estimates exceed the envelope formed by the filter indicated error deviations. In order to obtain good filter tuning, it is desired that the true RMS errors, as described by Eq (5-7), remain within the envelope formed by the $P_{jj}(t_i)$ values.

Since failure to model spatial correlations seriously degrades tracking performance, the majority of the simulations in this research contain the filter model of spatial correlations within the measurement noise with a strength that is twice the true noise strength. This value was not the result of tuning accomplished in this research, but rather, was chosen for use in comparison with the tuned BM filter previously developed. Tests were not made with respect to unmodelled spatial correlations in the filter containing the constant turn-rate acceleration model; however it can be expected to respond similarly since the measurement portions of both filters are identical. It should be noted that the signal to noise ratio (SNR), as defined by Eq (5-8), in these cases was a value of 20. Thus, the dependence of the GM and CTR filters on accurately modelling the spatial correlations of the background noise is in direct contrast to the results obtained for the BM filter at this SNR. Harnly and Jensen showed that the presence of unmodelled spatial correlations was important at low SNRs, but of little consequence when the SNR was as high as 10 or 20.

Therefore, in the tracking environment with a single hot-spot target image, the BM filter is much more robust (in regard to sensitivity to spatial correlations in the background noise) than either of the two filters developed in this research. As a result, when all filters are given the same spatial correlation model, the BM filter can be expected to perform best under conditions where the true correlations vary and/or are not modelled correctly.

6.3 Basic Filter Comparisons

This section will contain comparisons of the basic tracking performance of the filters with the first order Gauss-Markov acceleration model, the constant turn-rate acceleration model, as well as Harnly and Jensen's Brownian motion acceleration model. Initially, performance in the presence of a single hot-spot image with circular equal-intensity contours will be discussed. This is followed by the performance of the filters when the target image consists of the combination of three hot-spots, fixed in relation to one another, each of identical maximum intensity (SNR = 20), with each individually presenting circular equal-intensity contours. Elliptical contours were implemented in the cases involving dynamic image variations with the results discussed in section 6.4.

6.3.1 Single Hot-Spot Image. Trajectory 3 of Table 5.1 was used to compare tracking performance of the filters against the single hot-spot image. The image is characterized by a SNR of 20, a Gaussian glint dispersion parameter, σ_g^2, of value 3, and circular equal-intensity contours. The variance of the dynamics driving noise, σ_d^2, was either 160 or 300 where the two values were considered for tuning purposes. The trajectory itself is benign in the sense that the magnitude of the inertial acceleration is zero; however, there is significant target motion along the range unit vector (in inertial space) which is unmodelled by the filters examined in this research. When a dynamic image is being observed on the image plane (such results are discussed in section 6.4) motion along the range unit vector creates changing size effects as well as an unmodelled rotation of the tracker coordinate system as the target gets closer. These effects were not present in cases discussed in this section since they all utilized a static target image. However, what the filter is actually estimating, in either type of image, is the dynamics of the target in the plane perpendicular to the line of sight. Since portions of these parameters are dependent on range dynamics, these contributions are unmodelled by any of the filters' dynamics models. For example, the true planar acceleration perpendicular to the LOS includes a component that is proportional to the radial velocity of the target. This along with other unmodelled components can cause biases in the state estimates. The results of the overall performance of the individual filters is sumarized in Table 6.3.

Table 6.3 Time Averaged Error Statistics when Tracking a Single Hot-Spot Image

Filter Model	x Dynamics Position			y Dynamics Position		
	Peak Error	Mean Error	Std. Dev.	Peak Error	Mean Error	Std. Dev.
BM $\sigma_d^2 = 160$	.12	.00	.17	.12	.00	.17
GM $\sigma_d^2 = 160$	.291	.149	.509	-.404	-.249	.429
GM $\sigma_d^2 = 300$	.312	.162	.509	-.411	-.254	.429
CTR $\sigma_d^2 = 160$	.149	.090	.199	-.175	-.042	.229

Along with the plots in Figures F-1 through F-8, Table 6.3 indicates that Harnly and Jensen's adaptive filter, with its Brownian Motion acceleration model, is essentially unbiased in both channels with peak errors and standard deviations of less than 0.2 pixels. It should be noted, however, that when this run was made, the maneuver indicator that Harnly and Jensen implemented was triggered several times. The purpose of the maneuver indicator is to detect the beginning of a harsh maneuver and then to increase the Kalman filter gains directly and to reprocess the state estimate with the higher gains. Activation of the maneuver indicator can be identified by observing the sharp increases in the filter sigma estimates of the velocity states. Although use of this indicator is an ad hoc procedure, and not part of their basic filter model, the maneuver indicator should not have been triggered since no change in target dynamics occurred. It is felt, and will be discussed subsequently, that these activations are related to a sensitivity to the noise values generated through the use of pseudorandom codes.

In regard to the performance of the Gauss-Markov filter studied in this research, it can be seen from the entries in Table 6.3 and the plots of Figures D-21 through D-32 that the nonzero biases and error standard deviations on the order of 0.5 pixels in the position estimates cause the RMS position errors to exceed the filter sigma envelopes by as much as 0.2 pixels on a consistent basis. When this performance was observed for $\sigma_d^2 = 160$, attempts were made to tune the filter further. However, within the inverse covariance update (Eq (4-27)) the eigenvalues of $\underline{P}^{-1}$ corresponding to the dynamics position states are dominated by $\underline{H}^T\underline{R}^{-1}\underline{H}$. Thus, increasing σ_d^2 to 300 does not affect the one-sigma envelope or increase the gains associated with the position states (see plots in Figures D-33 through D-36). In fact, performance was slightly degraded by increasing σ_d^2. This indicates that further tuning of the position states should be done by varying $\underline{R}$ (σ_b^2). The biases present can be attributed, at least to some degree, to the motion of the target that is unmodelled by the filter.

In contrast, the constant turn-rate filter displays much smaller biases and standard deviations as indicated by the statistics in Table 6.3 and shown by the position plots in Figures E-1 through E-4 (x channel) and in Figures E-5 through E-6 (y channel). This improvement further indicates that the dynamics portion of the Gauss-Markov filter is responsible for a large degree of the error in that filter's estimates. The distinct difference in the two dynamics models is illustrated by examining the corresponding velocity and acceleration plots. The Gauss-Markov velocity plots are essentially unbiased while the acceleration plots display large biases. On the other hand, the constant turn-rate plots are characterized by just the opposite since this dynamics model is a much more accurate representation of the actual accelerations of the target image.

6.3.2 Three Hot-Spot Image. To contrast filter performance in the previous scenario, a much more complex target image was presented to the same three filters. The image itself is static except for the presence of background/measurement noise. The individual spots of the image are bivariate Gausian intensity distributions with circular equal-intensity contours each with glint dispersion parameter, σ_g^2, of value 2.0. This particular image is displayed in Figure 18 in the previous chapter. The trajectory used to test filter performance against this image is Trajectory 1 of Table 5.1 in which the target exhibits a constant velocity in the $-\vec{i}$ direction. The statistical results of the position estimates for the overall simulation are presented in Table 6.4 for each of the three filters. In all cases the variance of the filter dynamics driving noise, σ_D^2 was 300.

Table 6.4 Filter Performances with a Multiple Hot-Spot Image

Filter Model	x_D^+ Dymanics Position			y_D^+ Dynamics Position		
	Peak Mean Error	Time Ave Mean Error	Time Ave Stand. Dev.	Peak Mean Error	Time Ave Mean Error	Time Ave Stand. Dev.
BM	0.25	0.00	1.3	-0.81	-0.70	0.20
BM*	±0.70	0.00	0.95	-0.60	-0.40	0.37
GM	-0.46	-0.29	0.47	0.12	0.04	0.61
CTR	-0.23	-0.11	0.27	0.14	0.06	.37

*Maneuver indicator enabled

The plots corresponding to the results of the first case displayed in Table 6.4, for Harnly and Jensens adaptive filter, are shown in Figures F-9 through F-16. They indicate that this filter's typical performance is seriously degraded. Although the x channel position and velocity estimates are relatively unbiased, the standard deviation of the actual errors is over five to six times greater than the filter indicated errors. In contrast, the y channel exhibits a standard deviation similar to the previous scenario (results in Table 6.3) while the position estimate experienced a bias of about -0.7 pixels. The difference in behavior of the two channels can be attributed to the fact that almost all of the image plane dynamics is in the x direction. Thus there is likely to be more variation between the filter predicted x location and the true x location. Another characteristic of the filter's performance was its formulation of a reference profile corresponding to a single hot-spot image with a maximum intensity that was consistently equal to 40 (as opposed to the true value of 20 for each spot). In addition, the estimated one-sigma image dispersion estimates ranged anywhere from 5 to 6 in the velocity direction with an estimated aspect ratio typically from 3 to 4.

In an effort to see how activation of the maneuver indicator might affect filter tracking of the three spot image, a case was performed with the indicator enabled. The results in Table 6.4 (second set of entries) and the plots in Figures F-17 through F-24 display quite different characteristics than in the previous case. Although the standard deviations of the errors of the x channel position and velocity remained comparable; the mean error, which remained unbiased, exhibits much larger fluctuations (peaks as high as 0.7 pixels). In addition, the y channel position estimate displays a smaller bias of about -0.4 pixels, but with a larger standard deviation equal to 0.37 pixels. As in the previous case, the filter modelled the three spot image as a single spot image with an estimated maximum intensity estimate that was twice as large as the true value of any of the actual spots present. In addition, the size of the filter produced image was large enough to encompass the entire three spot image shown in Figure 18. In both cases the estimate appeared to oscillate back and forth (in the x direction) between a pair of spots while remaining more or less equidistant from the third (thus the bias in the y channel). In regard to the 10 Monte Carlo simulations conducted for each run, such motion can range from being totally "in phase" to being totally "out of phase." That is, when "in phase," the simulations follow the same error sequences in regard to sign of the errors. Thus, it appears as though more of the simulations of the second run are "in phase" than in the first case. Other than the maneuver indicator being enabled, the parameters of these runs were identical. Therefore, such behavior must have been caused by activation of the maneuver indicator, possibly during the acquisition stage of the simulations. In any event, serious degradation in filter tracking performance results when the target image does not resemble the single hot-spot image that the filter expects to see.

The performance of the Gauss-Markov filter in this scenario (case 5 of Table D.1) is indicated by the performance plots in Figures D-37 through D-46 and the statistical results given by the third set of entries in Table 6.4. There exists a bias of -0.29 pixels in the $\hat{x}_D$ dynamics estimate as well as sizeable standard deviations in both channels. As a result, the RMS tracking errors are consistently outside the envelope formed by the filter's estimate of its own errors. The major contribution to these large RMS errors are the standard deviations associated with the errors. The large standard deviations are in turn related to the fact that, in this particular case, the filter was not given a model of the spatial correlations in the measurement noise. As indicated earlier, failure to model such correlations by the GM and CTR algorithms can increase the standard deviations of the error by as much as 100%.

In contrast, the constant turn-rate filter, in a case where spatial correlatins were modelled, showed significantly improved performance over all the previous cases discussed involving the three hot-spot image (see the last set of entries in Table 6.4). This run is represented as case 13 with plotted results appearing in Figures E-17 through E-22. Although the RMS tracking errors remain at or within the filter sigma envelopes of both channels, the standard deviations are larger than typical values for this filter. This is so because of increased difficulty in detecting small translations of an image that is as broad with shallow variations such as in this particular image.

6.4 Performance vs Changing Target Dynamics

In this section, the results corresponding to filter performances during changes in target dynamics such as constant G and constant roll-rate maneuvers are presented. Trajectories 2 and 4 of Table 5.1 are used to describe the constant G maneuvers while Trajectory 1 is used to illustrate constant roll-rate maneuvers. In each case, except for the Brownian Motion filter developed by Harnly and Jensen, the dynamic form of the multiple hot-spot image was implemented so that the three spots varied in size, shape, and orientation with respect to each other on the image plane. The reference size of this image is set by the parameters described in section 5.4 and is displayed in Figure 19. In the cases with the BM filter, a single spot static image was implemented since it experiences too much difficulty in tracking multiple spot images.

6.4.1 Constant G Maneuver Performances. Table 6.5 displays the statistical results of the performances of the filters during the last two seconds of each engagement after transients had died out. The first set of entries in the table correspond to Harnly and Jensen's BM filter performance against a 2G pull-up (in the manner indicated by Trajectory 2) initiated at a time of two seconds after simulation initiation. The figures containing the plots of this performance are Figures F-25 through F-32. Although there is approximately a 1 pixel/sec bias throughout the x velocity estimate and a -1.5 pixel/sec bias in the y channel velocity after the start of the maneuver, the overall performance is very good. The position estimates are essentially unbiased both before and after the manuever initiation (neglecting transients) with standard deviations on the order of 0.2 pixels. However, it should be noted that this author had some difficulty in obtaining a run with

10 out of 10 successful tracking simulations of the 2G maneuver. Loss of track ocurred as much as 40 percent of the time during this maneuver due to sensitivity to the specific sequence of simulated noise values as mentioned earlier. Changing the seed of the pseudorandom code used to generate the noise values enabled the performance indicated in these plots; however, there is still a non-negligable probability of loss of track. This probability increases with the degree of dynamics involved in the maneuver, since no trials were found in which the filter maintained track on 10 out of 10 simulations of a 5G maneuver.

The next three sets of entries in Table 6.5 describe the results of the first order Gauss-Markov and the constant turn-rate filters' performances against the same 2G maneuver. Although the mean errors are markedly larger than those of the BM filter, it should be kept in mind that these filters are tracking a much more complex target image that is coming out of a roll maneuver prior to the 2G pull-up (the results of the BM filter in this table correspond to a static single spot image). The plots corresponding to the

Table 6.5 Constant G Maneuver Performances

Filter Model (Trajectory 3)	Case	x Dynamics Position			y Dynamics Position		
		Peak Mean Error	Time Avg Mean Error	Time Avg Stand. Dev.	Peak Mean Error	Time Avg Mean Error	Time Avg Stand. Dev.
BM 2G $\sigma_D^2=300$	Single Spot	0.15	0.05	0.17	0.30	0.01	0.20
GM 2G $\sigma_D^2=300$	9	0.378	0.184	0.148	-0.379	-0.247	0.163
CTR 2G $\sigma_D^2=300$	20	0.440	0.369	0.156	-0.075	-0.007	0.093
CTR 2G $\sigma_D^2=600$	21	0.403	0.325	0.159	0.079	0.002	0.094
CTR 2G $\sigma_D^2=1000$	22	0.313	0.226	0.139	-0.604	-0.495	0.376
CTR 2G $\sigma_D^2=5000$	23	0.435	0.334	0.168	0.183	0.063	0.133
(Trajectory 4)							
Bm 5G $\sigma_D^2=600$	Single Spot	0.20	0.10	0.20	0.15	0.00	0.18
GM 5G $\sigma_D^2=600$	10	0.31	0.174	0.132	-0.114	0.005	0.123
CTR 5G	24	0.274	0.207	0.136	-0.127	-0.016	0.124

GM 2G entry are shown in Figure D-77 through D-92. Similarly, the plots of the CTR filter are shown in Figures E-91 through E-96 and Figures E-97 through E-104. The results indicate more difficulty in tracking the x channel even though the maneuver is mainly in the y direction. This is so because of the particular image described in this simulation. The target is oriented such that the image is very smooth and broad in the x direction. That is, the intensity of the image only drops to the one-sigma values at the edges of the FOV in regard to the x direction. Thus the residual information due to shifts in the x direction is much less significant than that obtained by equivalent shifts in the y direction. In addition, the maximum derivatives of the intensity profile with respect to the x direction are also smaller, resulting in correspondingly lower entries in the linearized intensity function matrix. These results indicate that the CTR and GM filters perform essentially the same with respect to the x position estimate; however, there is significant improvement in the y position estimate of the CTR filter after transients from maneuver initiation have died out. This performance is also indicated by the cases that use a 5G manuever rather than a 2G within Trajectory 2. The GM filter never maintained track on this maneuver while the CTR filter required a dynamics driving noise variance of 5000 to achieve satisfactory performance in the y direction (see Table 6.5).

In contrast to Trajectory 2 constant G maneuvers were also conducted in a horizontal inertial plane as described by Trajectory 4. This type of maneuver has much less effect on target image dynamics in the y channel. The results of the last three sets of entries of Table 6.5 indicate this for the case of a 5G turn being conducted, with a variance of 600 for the dynamics driving noise. The corresponding plots are shown in Figures F-33 through F-40 for the BM filter, Figures D-93 through D-104 for the GM filter, and E-120 through E-135 for the CTR filter. The CTR filter no longer outperforms the GM filter since the image acceleration on longer resembles a constant turn-rate process.

6.4.2 Constant Roll-Rate Maneuvers. The most drastic variations of a target image occur when the target undergoes a rolling maneuver. In the case of a multiple hot-spot image such rapid variations can cause large errors in the filter produced reference image. In order to maintain more accurate shapes, more weight must be put on the current measurement frame when smoothing is accomplished in the data processing portion of the filter that is used to obtain the nonlinear intensity function, $\underline{h}[\underline{x}(t),t]$. However, an increase in the smoothing constant, as defined in Eq (2-5), also creates greater errors in the nonlinear intensity function due to the additional effects of background/measurement noise. Thus a tradeoff analysis of the two effects is required.

Table 6.6

Performance vs Constant Roll-Rate Maneuvers

Filter Model	Roll Rate (Rev/Sec)	x_D Channel			y_D Channel		
		Peak Mean Error	Time Avg Mean Error	Time Avg Stand. Dev.	Peak Mean Error	Time Avg Mean Error	Time Avg Stand. Dev.
GM $\alpha=.1$	0.5	0.272	0.227	0.160	-0.212	-0.039	0.126
GM $\alpha=.1$	1.0	-0.233	-0.042	0.190	-0.148	-0.024	0.117
GM $\alpha=.1$	2.0	-0.223	-0.002	0.189	-0.173	-0.048	0.115
CTR $\alpha=.1$	0.5	0.380	0.210	0.153	-0.185	-0.055	0.126
CTR $\alpha=.1$	1.0	0.402	0.239	0.167	-0.186	-0.071	0.117
CTR $\alpha=.2$	0.5	0.539	0.341	0.151	-0.284	-0.100	0.163
CTR $\alpha=.2$	1.0	0.644	0.377	0.140	-0.240	-0.091	0.136
CTR $\alpha=.5$	0.5	0.935	0.497	0.178	-0.666	-0.340	0.197
CTR $\alpha=.5$	1.0	2.214	1.057	0.260	-1.757	-0.896	0.295

The first three entries of Table 6.6 correspond to cases 7, 8, and 9 of Table D.1 in which the GM filter tracked a target along Trajectory 1 with roll-rates of 0.5, 1.0 and 2.0 revolutions per second. Examining either channel of the plots of case 7 (shown in Figures D-47 through D-62), evidence of distinct oscillations in the filter mean errors is present both before and after measurement updates. These oscillations occur at a frequency that is twice the roll-rate of the target since the target image is identical after every 180 degrees of revolution. Also note that the oscillations in the x channel position, velocity, and acceleration errors diminish in magnitude until they vanish by the end of the simulation. This is due to the location of the target as it proceeds along this trajectory. At the end of the simulation the target is located at point where it is travelling in a direction that is exactly perpendicular to the line of sight from the sensor. Thus, the motion of the image spots due to the roll is confined entirely to the y axis. In contrast, the y channel errors oscillate throughout the entire simulation. As the target roll-rate is increased from 0.5 to 1.0 (see plots in Figures D-63 through D-68) and then to 2.0 revolutions per second (see plots in Figures D-69 through D-76), the oscillations display much more rapid fluctuations (i.e., as the roll-rate increases, there is a corresponding increase in the error fluctuation rate). The statistical results of Table 6.6 indicate that the corresponding errors decrease as this occurs. This "improved" performance is due to the fact that the errors in the reference image are such that they begin to compensate for each other in regard to the position translational errors produced.

Since the CTR filter was based on a better model overall, it was used to test the effects of roll-rate variations vs smoothing constant variations. The results are presented as the last six sets of entries of Table 6.6, not all of which were plotted. These statistics indicate a consistent increase in all errors as the smoothing constant was increased from 0.1 to either 0.2 or 0.5. This occurred for roll-rates of either 0.5 rev/sec or 1.0 rev/sec. Although such trends might not be expected at first glance, part of the increases can be attributed to the additional noise effects as smoothing is reduced (by increasing the smoothing constant). It appears as though the remainder of the error increases can be related to the previous result, in which if the roll-rate is large relative to the smoothing constant, and the errors are cyclic, the resulting rapid fluctuations in the position errors are actually reduced in magnitude. Thus of the values of alpha studied in this research, 0.1 yields the best performance. However, further studies should be done on this parameter; since a value of 0.05 might achieve better performance, especially for slowly varying targets.

6.5 Dynamic Driving Noise Estimation

The intent of dynamic driving noise estimation was to enable the filter to adapt to changes in target dynamics. When the estimator is allowed to proceed unchecked, and the target does not change dynamics, the eigenvalues of the Q_{FD} matrix decreased until they eventually became negative (especially those related to the position states). Since such behavior is not appropriate, means of bounding the Q_{FD} matrix were investigated. Initially, the diagonal terms were simply increased to values of 0.1 and the corresponding row and column set to zero if that particular diagonal element had decreased below zero as the result of the estimation process described in section 4.7. This resulted in performance that failed to show significant improvement even in tracking benign trajectories such as Trajectory 1 of Table 5.1 (due to the dominance of $H^T R^{-1} H$ term in the inverse covariance update). Instead, the filter lost track when a 2G maneuver was initiated since even the eigenvalues of the Q_{FD} matrix corresponding to velocity and acceleration decreased to the 0.1 lower bound. As a result correspondingly larger values were chosen for the lower bounds of the diagonal terms related to the velocity and acceleration (the values chosen were equivalent to those obtained on the diagonal for a constant Q_{FD} matrix with dynamics driving noise strength of 160). When the estimated values of the diagonal terms decreased below these values, the corresponding row and column were scaled upward while maintaining the same correlation coefficients (i.e., each term in a row and column to be scaled is multiplied by the square root of the ratio of the desired lower bound to the estimated diagonal term). Such a bounding scheme enabled the filter to maintain track through the onset of 2G and 5G maneuvers. However, before the simulations were complete, negative eigenvalues developed in both the $\underline{P}^+$ and $\underline{P}^-$ matrices. The diagonal terms of these matricies were then bounded to 0.01, 1.0, and 10.0 for the diagonal terms related to position, velocity, and acceleration respectively. This enabled the filter to maintain track throughout 2G maneuvers. However, little benefit was obtained for all the extra computation involved. In fact, a case equivalent to case 20 (in regard to filter parameters), the estimator actually increased the mean position errors by about 0.05 to 0.1 pixels in both channels. In general, however, position estimates were not affected substantially due to the dominance of the $\underline{H}^T \underline{R}^{-1} \underline{H}$ term in the covariance update equations.

VII. Conclusions and Recommendations

7.1 Conclusions

The conclusions discussed in this section are directly related to the results presented in Chapter VI. In general, the characteristics of the target image being tracked, in addition to the algorithms used to generate a reference profile of this image, have a significant effect on filter tracking performance. This is true even in benign trajectories.

As a basis for comparison, the Gauss-Markov and constant turn-rate filters developed in this research were contrasted with the previously developed Brownian Motion filter (Ref: 5). The Gauss-Markov and constant turn-rate filters, which used the data processing algorithm of Figure 1 (i.e., no a priori information is assumed about the image intensity distribution), reliably generated reference intensity profiles that corresponded well with true profiles. This performance was evident even when the algorithm was confronted by a dynamic image that corresponds to realistic projections of a "multiple hot-spot" target onto the FLIR image plane. In contrast, the Brownian Motion filter assumed a priori knowledge of the analytic form of these intensity functions.

When the Brownian Motion filter is faced with multiple hot-spot images such that its intensity model is no longer accurate, performance is seriously degraded. In contrast, the Gauss-Markov and constant turn-rate filters displayed good tracking performance even when faced with drastic image variations inherent in constant G and constant roll-rate maneuvers. Thus the need for pattern recognition techniques, such as those used in this research, is apparent.

The constant turn-rate filter significantly outperformed the Gauss-Markov filter in regard to errors in both directions on the FLIR image plane. This is to be expected since the dynamics model used in the constant turn-rate model is a much truer representation of typical target accelerations as projected onto the FLIR image plane. When the target dynamics are no more representative of a constant turn-rate process than a first order Gauss-Markov process, the two filters performed equivalently. However, biases on the order of 0.1 pixels appeared in many of the simulations for both filters. Such biases can be attributed to inaccuracies in the phase information used within the data processing algorithm and/or unmodelled dynamics of the target in the inertial reference frame. The latter involves unmodelled motion along the range unit vector as well as size and shape effects when a dynamic image is involved.

Along this line, tracking performance is somewhat sensitive to the smoothing constant and the rate of image variations during maneuvers such as constant roll-rates. Such maneuvers cause error oscillations (at a frequency of twice the target roll-rate) when image symmetry is disrupted by portions of the target being obscured. Increasing the value of the smoothing constant above 0.1 does not benefit performance due to additional noise effects. That is, when the smoothing constant is increased, more weight is put on the current measurement received; thus, the reference image becomes more susceptible to errors due to the addition of background/measurement noise.

The Gauss-Markov and constant turn-rate filters were found to be less robust in regard to spatial correlations in the background noise than the Brownian Motion filter. Thus, in scenarios where these correlations vary or are mismodelled, the Gauss-Markov and constant turn-rate filters should experience nonnegligible errors even in high signal to noise environments. An additional effect of modelling spatial correlations by the filter is the dominance of the $\underline{H}^T\underline{R}^{-1}\underline{H}$ term in the inverse covariance update equations. This requires that tuning of the filter position states be done by changing the variance of the filter assumed measurement noise, σ_b^2 , rather than that of the dynamics driving noise, σ_d^2 .

7.2 Recommendations

Further study is recommended in order to extend the research accomplished in this effort as well as to refine some of the tracking performances displayed by the first order Gauss-Markov and constant turn-rate extended Kalman filters. Specifically, items of further research should include, but are not limited to, the following:

- Perform additional filter tuning by varying σ_b^2, the variance of the background noise, in order to determine whether bias errors can be reduced. Also consider various truth model values of this parameter in order to create different signal to noise ratio environments.

- Include estimates of range/range-rate parameters in the dynamics models in order to account for bias errors caused by the unmodelled motion in the inertial reference frame.

- Examine the performance with smoothing constants less than 0.1, especially for slowly varying target images.

- Investigate implementation of optical techniques for processing the measurement data in portions of the algorithm displayed in Figure 1.

- Consider adaptive expansion of the field of view to maintain track on harshly maneuvering targets at close ranges.

- Determine the effects of radiation emitted from the target due to the incidence of the high energy laser beam.

- Comparison of the algorithms studied in this research with an enhanced correlator/linear Kalman filter tracker on the basis of performance and computational loading in order to determine which is better suited for practical implementation.

- Examine the "crossover" effects introduced as the target flies by at minimum range and maximum angular rates. Implement trajectories that go past this point in order to observe filter recovery.

- Determine robustness to real world environments that differ strongly from the filter design models.

Bibliography

1. Bedworth, David D. *Industrial Systems Planning, Analysis, Control*. New York: The Ronald Press Company, 1973.

2. Cooley, J.W., P.A. Lewis and P.D. Welch, "Historical Notes on the Fast Fourier Transform," *IEEE Transactions on Audio and Electro-Acoustics,* Vol. AU-15, No. 2, pp 76-79, June 1967.

3. Flynn, Patrick M., "Alternative Dynamics Models and Multiple Model Filtering for a Short Range Tracker, "M.S. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1981.

4. Goodman, Joseph W., *Introduction to Fourier Optics*. San Francisco: McGraw-hill Book "Company, 1968.

5. Harnly, Douglas A. and Robert L. Jensen, "An Adaptive Distributed-Measurement Extended Kalman Filter for a Short Range Tracker," M.S. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1979.

6. Hogge, C.B. and R.R. Butts, "Frequency Spectra for the Geometric Representation of Wavefront Distortions Due to Atmospheric Turbulence," *IEEE Transactions on Antennae and Propagation*, Vol. AP-24, No. 2, March 1976.

7. Maybeck, Peter S. *Stochastic Models, Estimation and Control Volume I*. New York: Academic Press Incorporated, 1979.

8. Maybeck, Peter S. *Stochastic Models, Estimation and Control Volume II*. New York: Academic Press Incorporated, 1982.

9. Maybeck, Peter S. and Douglas A. Harnly and Robert L. Jensen, "Robustness of a New Infrared Target Tracker, *Proceedings of IEEE National Aerospace and Electronics Conference*, Dayton, Ohio, pp 639-644, May 1980.

10. Maybeck, Peter S. and Daniel E. Mercier, "A Target Tracker Using Spatially Distributed Infrared Measurements," *IEEE Transactions on Automatic Control*, Vol AC-25, No. 2, pp 222-225, April 1980.

11. Maybeck, Peter S., William H. Worsley and Patrick M Flynn, "Investigation of Constant Turn-Rate Dynamics Models in Filters for Airborne Vehicle Tracking," *Proceedings of IEEE National Aerospace and Electronics Conference*, Dayton, Ohio, pp 896-903, May 1982.

12. Mercier, Daniel E., "An Extended Kalman Filter for use in a Shared Aperature Medium Range Tracker," M.S. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1978.

13. Merrit, Paul H., "Beam Intensity Calculations for Jittered Beams," AFWL-TR-78-174, Air Force Weapons Laboratory, Kirtland AFB, New Mexico.

14. Oppenheim, Alan V. *Applications of Digital Signal Processing*, New Jersey: Prentice-Hall Incorporated, 1978.

15. Oppenheim, Alan V. and Ronald W. Shaffer, *Digital Signal Processing*. New Jersey: Prentice-hall Incorporated, 1975.

16. Rogers, Steven K., "Enhanced Tracking of Airborne Targets Using Forward Looking Infrared Measurements,: M.D. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1981.

17. Singletery, James Jr., "Adaptive Laser Pointing and Tracking Problem," M.S. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1980.

18. The Analytic Sciences Corporation. *Advanced Adaptive Optics Control Techniques*. TR-966-1. Prepared for the Air Force Weapons Laboratory, Kirtland AFB, New Mexico, January 6, 1978.

19. Worsley, William H., "Comparison of Three Extended Kalman Filters for Air-to-Air Tracking," M.S. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1980.

Appendix A

Truth Model $\underline{Q}_{TD}$ Matrix Derivation

This appendix is concerned with the derivation of the non-zero elements of the dynamic driving noise matrix of the truth model. As defined in Chapter III, $\underline{Q}_{TD}$ is given by

$$\underline{Q}_{TD} = \int_{t_i}^{t_{i+1}} \underline{\Phi}_T(t_{i+1},\tau)\underline{G}_T\underline{Q}_T\underline{G}_T^T\underline{\Phi}_T^T(t_{i+1},\tau)d\tau \tag{A-1}$$

where

$$\underline{Q}_T = \begin{bmatrix} Q_A & 0 \\ 0 & Q_A \end{bmatrix} \qquad \underline{G}_T = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ G1 & 0 \\ G2 & 0 \\ G3 & 0 \\ 0 & G1 \\ 0 & G2 \\ 0 & G3 \end{bmatrix}$$

and

$$\underline{\Phi}_T(t_{i+1},\tau) = \left[\begin{array}{cc|c|c} 1 & 0 & \multirow{2}{*}{$\underline{0}$} & \multirow{2}{*}{$\underline{0}$} \\ 0 & 1 & & \\ \hline \multicolumn{2}{c|}{\underline{0}} & \underline{\Phi}_{Ax}(t_{i+1},\tau) & \underline{0} \\ \hline \multicolumn{2}{c|}{\underline{0}} & \underline{0} & \underline{\Phi}_{Ay}(t_{i+1},\ ) \end{array}\right]$$

in which $\underline{\Phi}_{Ax} = \underline{\Phi}_{Ay}$, given by Eq (3-9), represent the state transition matricies for the atmospheric effects on target position on the FLIR image plane in the x and y directions respectively.

Thus

$$\underline{Q}_{TD} = \int_{t_i}^{t_{i+1}} \underline{\mathcal{L}}(t_{i+1},\tau)d\tau \tag{A-2}$$

where $\underline{\mathcal{L}}$ is an 8x8 symmetric block diagonal matrix. Each nonzero element of $\underline{\mathcal{L}}$ is a function of $(t_{i+1} - \tau)$ such that $\underline{\mathcal{L}}(t_{i+1},\tau) = f(t_{i+1} - \tau)$. Making the substitutions $u = t_{i+1} - \tau$ and $du = -d\tau$, the old limits of integration, $t_i \rightarrow t_{i+1}$, become the new limits, $\Delta t \rightarrow 0$, when the sampling time is held constant. As a result, Eq (A-2) becomes

$$\underline{Q}_{TD} = \int_{\Delta t}^{0} \underline{\mathcal{L}}(u)(-du) = \int_{0}^{\Delta t} \underline{\mathcal{L}}(u)du \tag{A-3}$$

$$Q_{TD} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & Q1 & Q2 & Q3 & 0 & 0 & 0 \\ 0 & 0 & Q2 & Q4 & Q5 & 0 & 0 & 0 \\ 0 & 0 & Q3 & Q5 & Q6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & Q1 & Q2 & Q3 \\ 0 & 0 & 0 & 0 & 0 & Q2 & Q4 & Q5 \\ 0 & 0 & 0 & 0 & 0 & Q3 & Q5 & Q6 \end{bmatrix} \tag{A-4}$$

The equivalence of the two nonzero blocks in Eq (A-4) is due to the independence of the model in the FLIR x and y directions. Each of the six different nonzero elements in Q_{TD} represents a single component of the matrix integral in Eq (A-3). The exact results are as follows:

$$Q1 = \int_0^{\Delta t} G1^2 e^{-2Au} du = (G1^2/2A)(1-e^{-2A\Delta t})$$

$$Q2 = \int_0^{\Delta t} (G1\ G2 + u\ G1\ G3) e^{-(A+B)u} du$$

$$= [G1^2/(A+B)][(1-e^{-(A+B)\Delta t})(-2B/(A+B))-(A-B)\Delta t e^{-(A+B)\Delta t}]$$

$$Q3 = \int_0^{\Delta t} G1\ G3\ e^{-(A+B)u} du = [(A-B)/(A+B)]\ G1^2[1-e^{-(A+B)\Delta t}]$$

$$Q4 = \int_0^{\Delta t} (G2^2 + 2\ G2\ G3\ u + G3^2u^2) e^{-2Bu} du$$

$$= (G1^2/2B)[(1-e^{-2B\Delta t})(1-(A-B)/B+(A-B)^2/2B^2$$

$$+ (A-B)(2-(A-B)/B)\Delta t e^{-2B\Delta t}-(A-B)^2\Delta t^2 e^{-2B\Delta t}]$$

$$Q5 = \int_0^{\Delta t} (G2\ G3 + u\ G3^2)e^{-2Bu}du$$

$$= [G1^2(A-B)/2B][(1-e^{-2B\Delta t})(A-3B)-(A-B)\Delta t e^{-2B\Delta t}]$$

$$Q6 = \int_0^{\Delta t} G3^2 e^{-2Bu}du = [(A-B)^2 G1^2/2B](1-e^{-2B\Delta t})$$

where A,B are the break frequencies of the shaping filter atmospherics model and,

$$G1 = \frac{KAB^2}{(A-B)}$$

$$G2 = -G1$$

$$G3 = (A-B)G1$$

which are the appropriate values for representation of the truth model atmcspherics in the Jordan canonical state space form (Ref: 12).

Appendix B

Derivation of the Constant Turn-rate Filter's $\underline{F}$ and Φ_F Matrices

The derivation of the linearized $\underline{F}$ matrix for the CTR filter is not as simple as it was for the first order Gauss-Markov filter. Since the dynamics model of the CTR filter is nonlinear, the point of linearization is dependent on the current value of the state vector and cannot be precomputed. Specifically, Eq(B-1) is used to calculate $\underline{F}$.

$$\underline{F}(t_i) = \left. \frac{\partial \underline{f}[\underline{x}(t),t]}{\partial \underline{x}} \right|_{\underline{x}=\hat{\underline{x}}(t_i^+)} \qquad \text{(B-1)}$$

The nonlinear function, $\underline{f}[\underline{x}(t),t]$, is as defined previously in Eq(4-14). Thus the $\underline{F}$ matrix has the form

$$\underline{F}(t_i) = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & F_1 & F_2 & F_3 & F_4 & 0 & 0 \\ 0 & 0 & F_5 & F_6 & F_7 & F_8 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & F_9 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & F_{10} \end{bmatrix} \qquad \text{(B-2)}$$

where

$$F_1 = -(\omega/A_1)(A_2 - 4\omega x_3^2 + 2x_3x_6)$$

$$F_2 = 2x_3\omega(x_5A_1 + 2x_4A_2)/A_1^2$$

$$F_3 = 2x_3x_4\omega/A_1$$

$$F_4 = -2x_3^2\omega/A_1$$

$$F_5 = -2x_4\omega(x_6A_1 - 2x_3A_2)/A_1^2$$

$$F_6 = -\omega[\omega - 2x_4(x_5A_1 + 2x_4A_2)/A_1^2]$$

$$F_7 = 2x_4^2\omega/A_1$$

$$F_8 = -2x_3x_4\omega/A_1$$

$$F_9 = F_{10} = -1/\tau_A$$

in which

$$A_1 = x_3^2 + x_4^2$$

$$A_2 = x_3x_6 - x_4x_5$$

$$\omega = A_2/A_1$$

τ_A = correlation time of the atmospheric jitter with the components of the state vector written as:

$$\underline{x} = [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8]^T$$

Once the $\underline{F}(t_i)$ matrix is determined, the state transition matrix can be evaluated using the quasi-static approximation:

$$\underline{\Phi}_F(t_{i+1},t_i) = \exp(\underline{F}(t_i)\Delta t) \tag{B-3}$$

The exact closed form of the two atmospheric terms can be determined easily; however, the remainder of Eq. (B-3) is approximated as shown by Eq. (B-4).

$$\underline{\Phi}_F(t_{i+1},t_i) = \begin{bmatrix} 1 & 0 & A & 0 & B & 0 & 0 & 0 \\ 0 & 1 & 0 & A & 0 & B & 0 & 0 \\ 0 & 0 & 1 & 0 & A & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & A & 0 & 0 \\ 0 & 0 & C_1 & C_2 & C_3 & C_4 & 0 & 0 \\ 0 & 0 & C_5 & C_6 & C_7 & C_8 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & D & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & D \end{bmatrix} \tag{B-4}$$

where

$A = \Delta t$

$B = \Delta t^2/2$

$C_1 = \Delta t F_1$ $C_2 = \Delta t F_2$

$C_3 = \Delta t F_3 + 1$ $C_4 = \Delta t F_4$

$C_5 = \Delta t F_5$ $C_6 = \Delta t F_6$

$C_7 = \Delta t F_7$ $C_8 = \Delta t F_8 + 1$

$D = e^{-\Delta t/\tau_A}$

Appendix C

Generation of a White Gaussian Noise Process

Since white Gaussian noises are present in many of the stocastic processes modelled in this research, it is useful to know how realizations of such noises can be generated. For instance, a discrete-time white Gaussian noise process with zero mean and a given variance can be obtained through the use of pseudorandom codes. Specifically, consider a psuedorandom code that produces samples of a scalar random variable, $f_x(\lambda)$, that is uniformly distributed between the values zero and one.

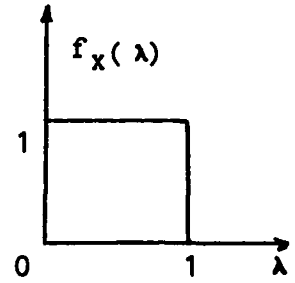


Figure C-1. Uniformly Distributed Random Variable

The mean of this distribution is given by

$$\mu\lambda = \int_{-\infty}^{\infty} f_x(\lambda)d\lambda = 1/2 \tag{C-1}$$

and the variance by

$$\sigma_\lambda^2 = \int_{-\infty}^{\infty} (\lambda-1/2)^2 f_x(\lambda)d\lambda = 1/12 \tag{C-2}$$

Now, if M independent calls are made to such a routine to produce the realizations, ζ_i, for i=1,M; which are then added together forming,

$$\Omega_j = \sum_{i=1}^{M} \zeta_i \qquad \text{(C-3)}$$

the result represents a realization of a new random variable with a mean of (M/2) and a variance of (M/12). The Central Limit Theorem states that the resulting distribution approaches a Gaussian as M grows without bound. In fact, it has been indicated that a Gaussian approximation becomes reasonable for an M greater than three (Ref: 7, p 109). As a result, a discrete realization of a Gaussian distribution of zero mean and unity variance can be had be choosing M equal to twelve and forming

$$\underline{\varepsilon}_j = \Omega_j - 6 \qquad \text{(C-4)}$$

where Ω_j is now the sum of twelve independent calls to the pseudorandom code. Finally, the realization of a white, zero mean, unity variance, Gaussian noise vector can be formed by ordering separate realizations of the scalar ε_j's into a vector of appropriate length.

Apendix D

This appendix contains the plotted output for the cases studied that involve the first order Gauss-Markov filter. Each plot is labled with its corresponding case number (1-11). The values of the parameters used in each of these cases are listed in Table D.1.

Table D.1a Parameters of Cases with GM Filter

Parameter	Case 1	2	3	4	5	6	7	8	9	10	11
Model	GM	GM	GM	GM	GM	GM	GM	GM	GM	GM	GM
Trajectory	1	1	2	2	1	1	1	1	3	4	1
REVRT	0	0	0	0	0	0.5	1.0	2.0	0.5	0.5	0
NG	0	0	0	0	0	0	0	0	2	5	0
ITARG	2	2	2	2	0	1	1	1	1	1	2*
VARDF	300	300	160	300	300	300	300	300	300	600	300
SIGAT	.141	.141	.141	.141	.141	.141	.141	.141	.141	.141	.141
VARAF	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
VARM	1	1	1	1	1	1	1	1	1	1	1
Filter	2	2	2	2	2	2	2	2	2	2	2
COV	2	2	3	3	2	-	-	-	-	-	2
ALPHA	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
VXO	-1000	-1000	-700	-700	-1000	-1000	-1000	-1000	-1000	-1000	-1000
VYO	0	0	-300	-300	0	0	0	0	0	0	0
VZO	0	0	500	500	0	0	0	0	0	0	0
RFIL	Y	N	Y	Y	N	Y	N	N	Y	Y	N

* Uses Subroutine SINGLE

Table D.1b Parameters Held Constant

Parameter	Value
IMAX	20
NF	0
NZ	0
NFRMS	150
NRUNS	10
(XO, YO, ZO)	(5000, 500, 20000)
(AXO, AYO, AZO)	(0, 0, 0)
SIGVO	(4.0, 1.5, 1.5)
SIGPVO	(1.0, 1.0, 1.0)
DELV	(.64, -.86, -.86)
DELPV	(0.0, 2.5, -2.5)
TDF	1.5
TAF	0.07072

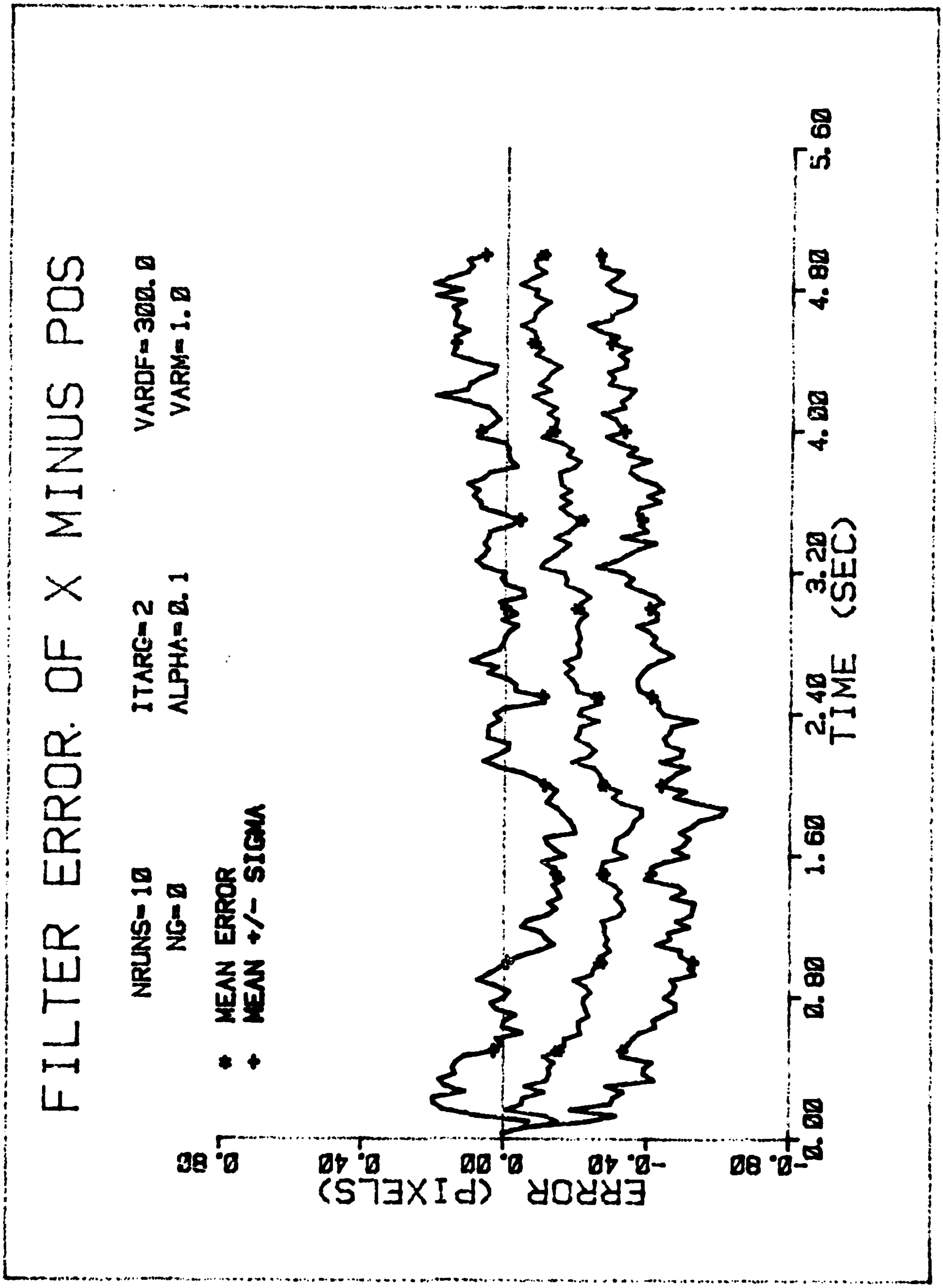


Figure D-1 Case 1 GM Performance Plot

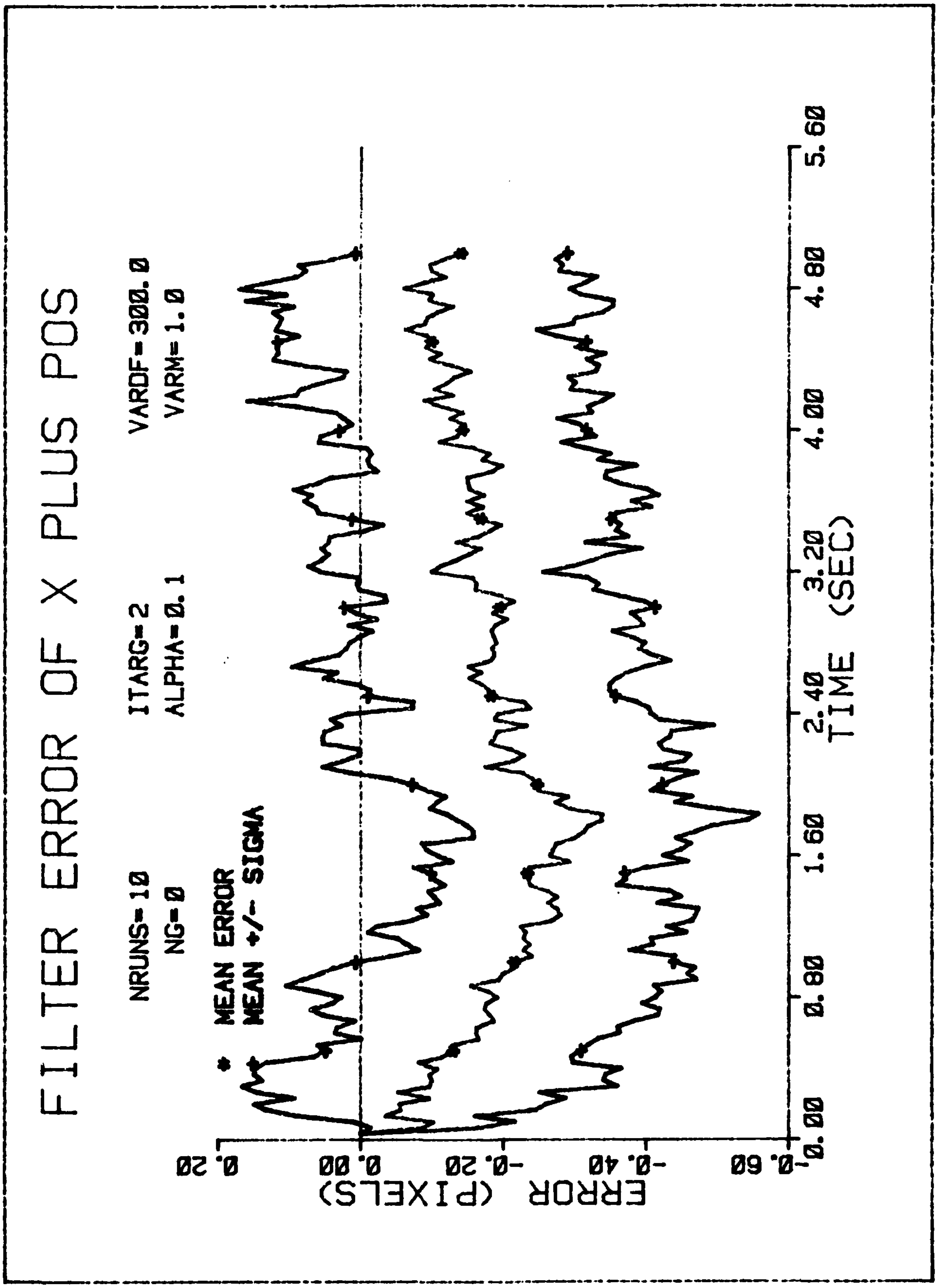


Figure D-2 Case 1 GM Performance Plot

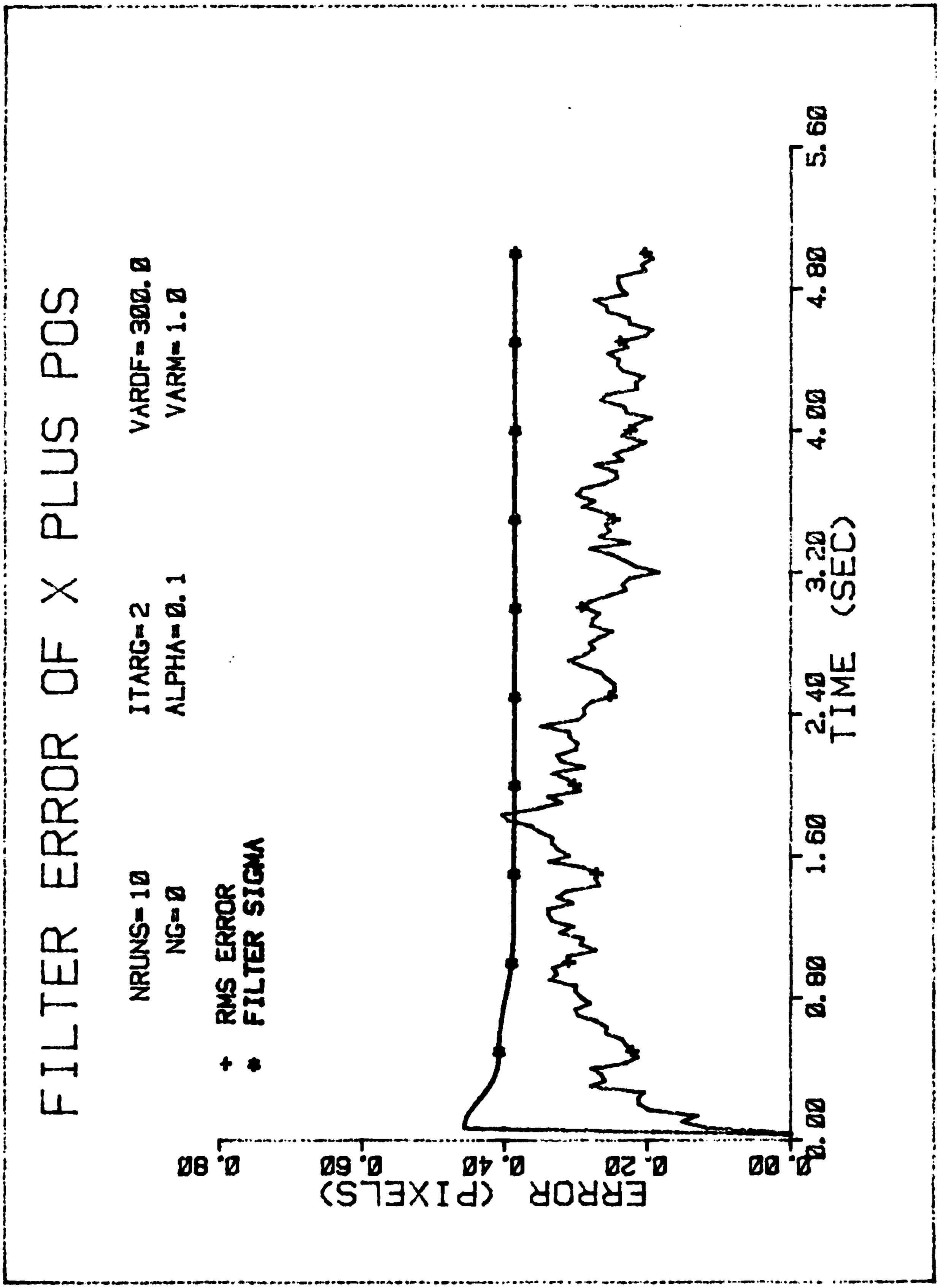


Figure D-3 Case 1 GM Performance Plot

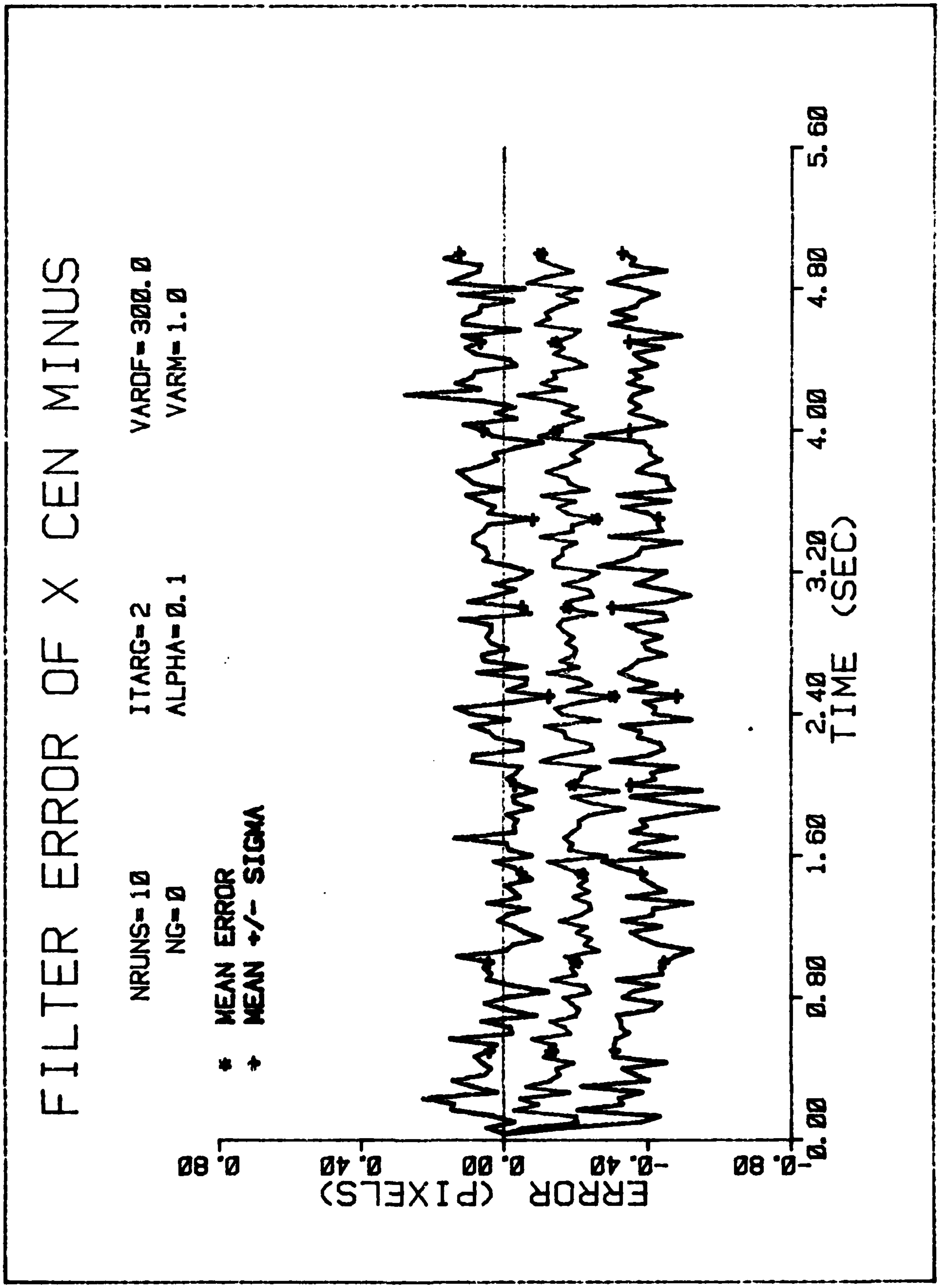


Figure D-4 Case 1 GM Performance Plot

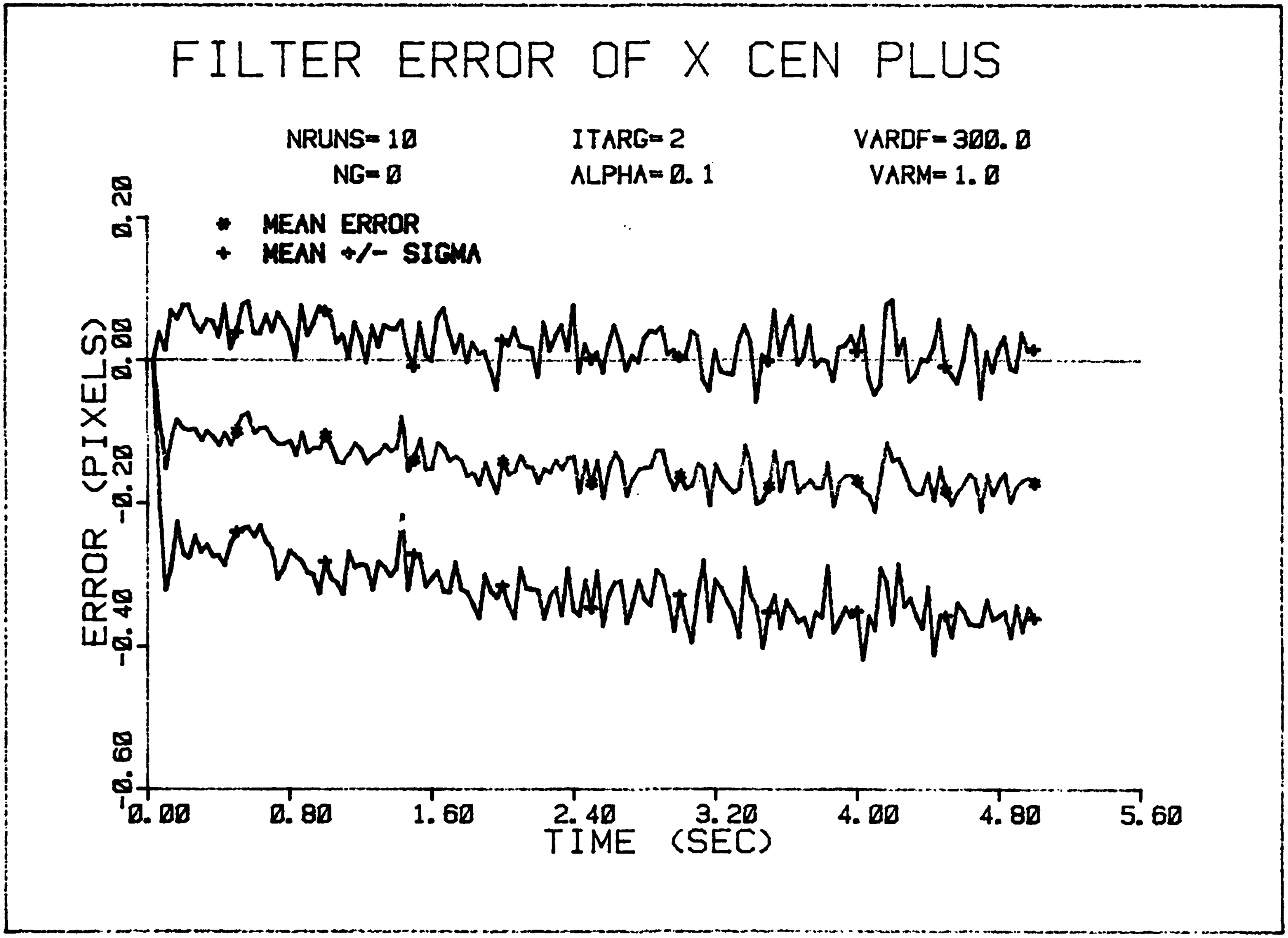


Figure D-5 Case 1 GM Performance Plot

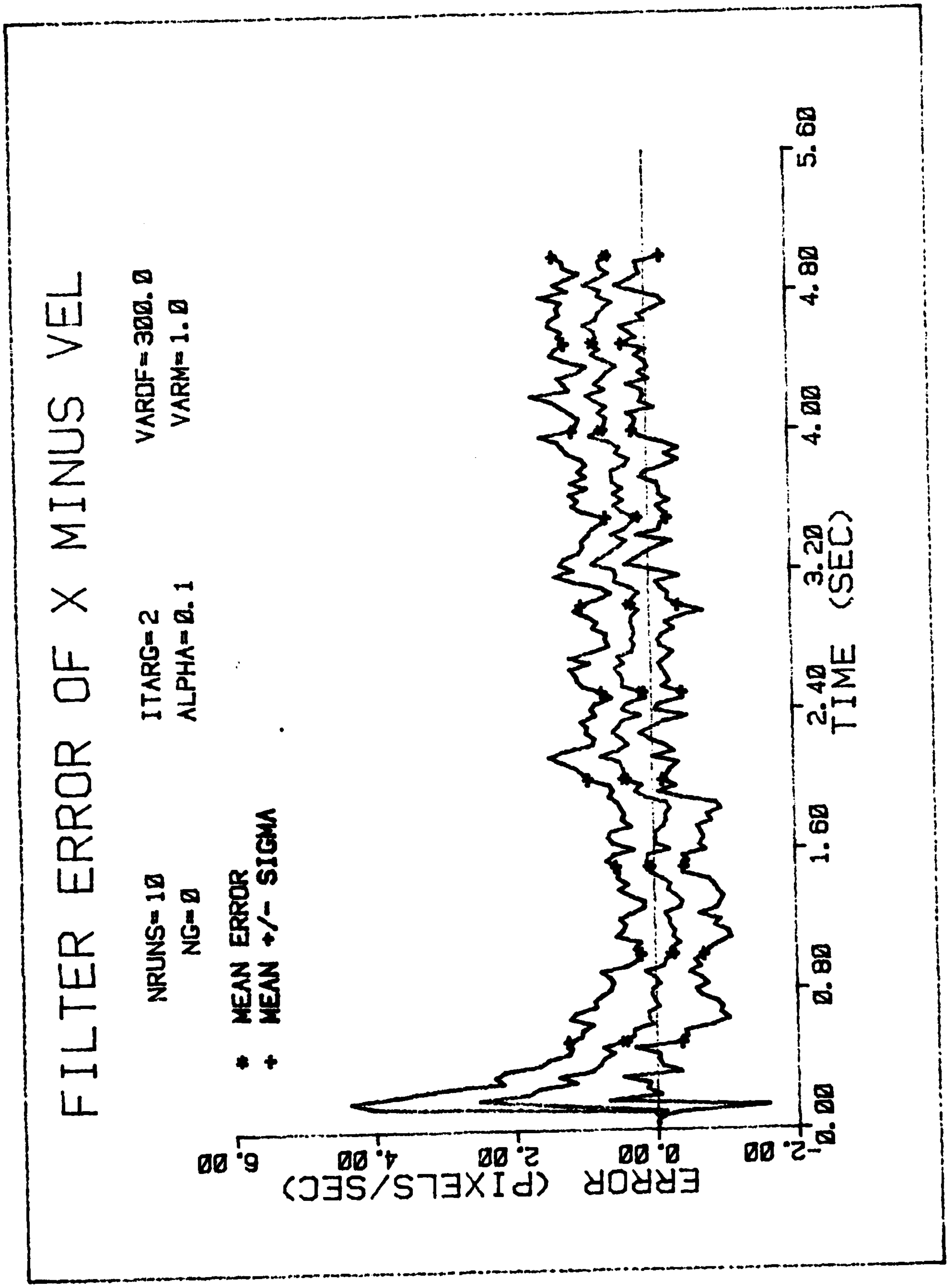


Figure D-6 Case 1 GM Performance Plot

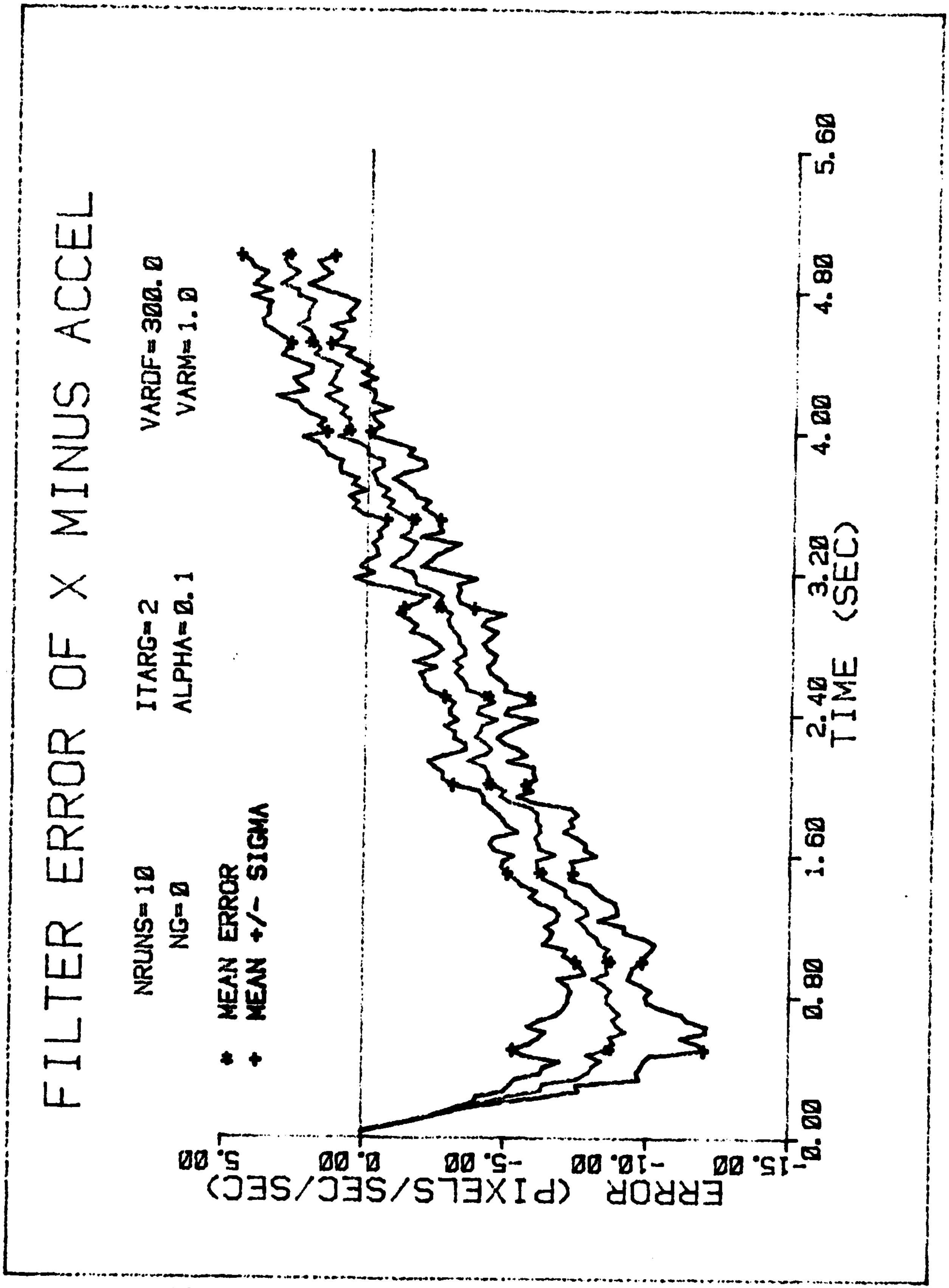


Figure D-7 Case 1 GM Performance Plot

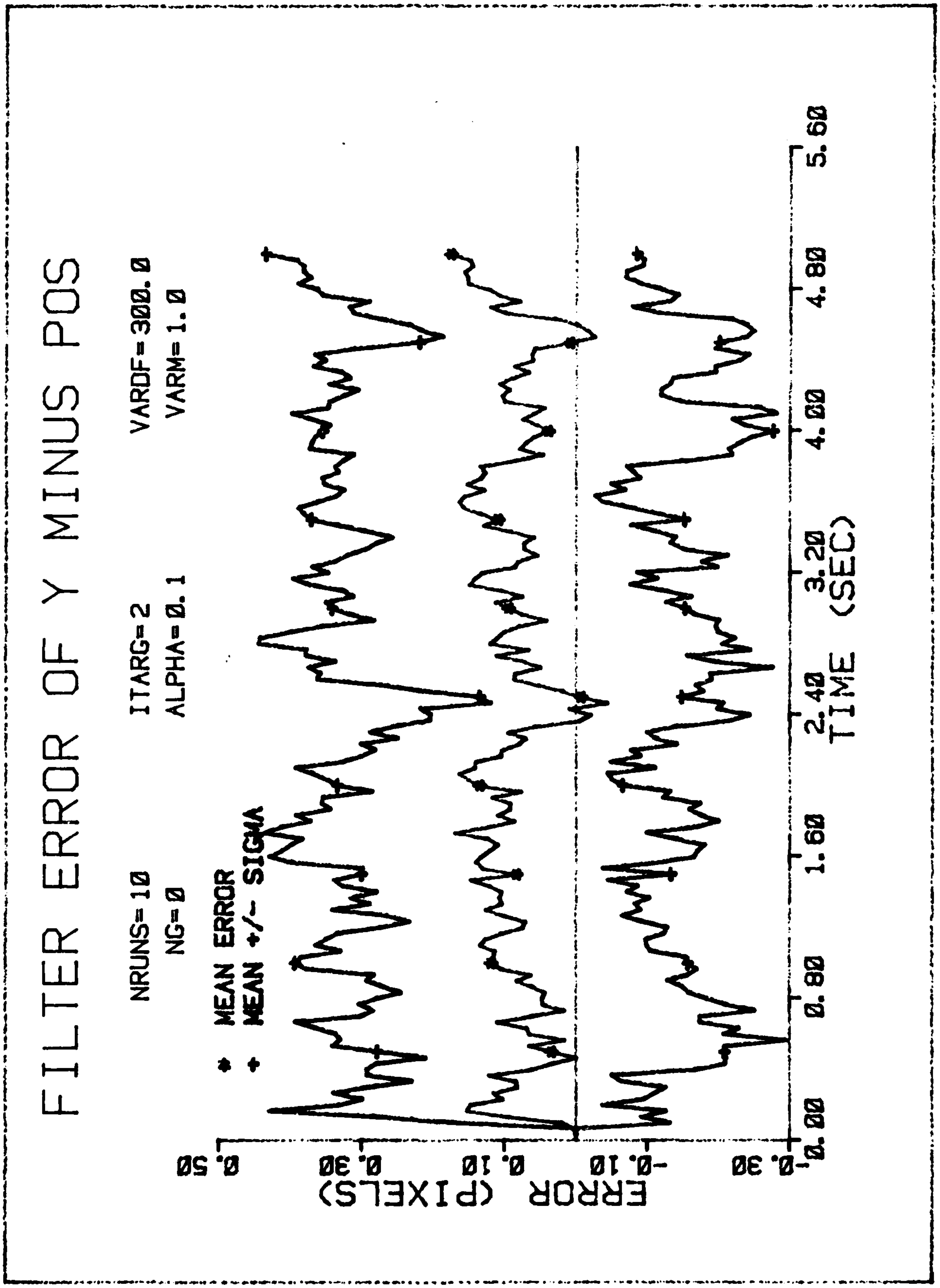


Figure D-8 Case 1 GM Performance Plot

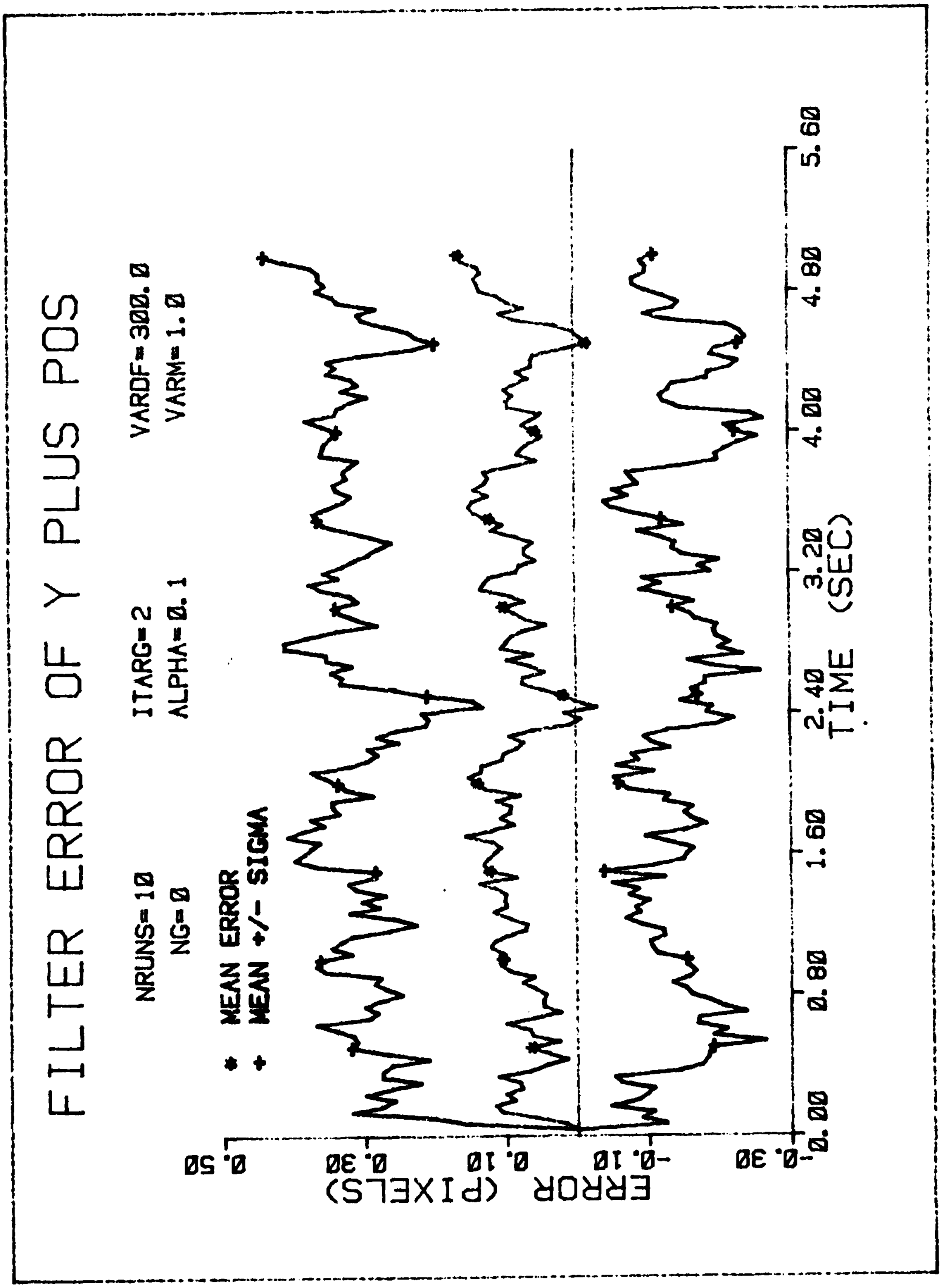


Figure D-9 Case 1 GM Performance Plot

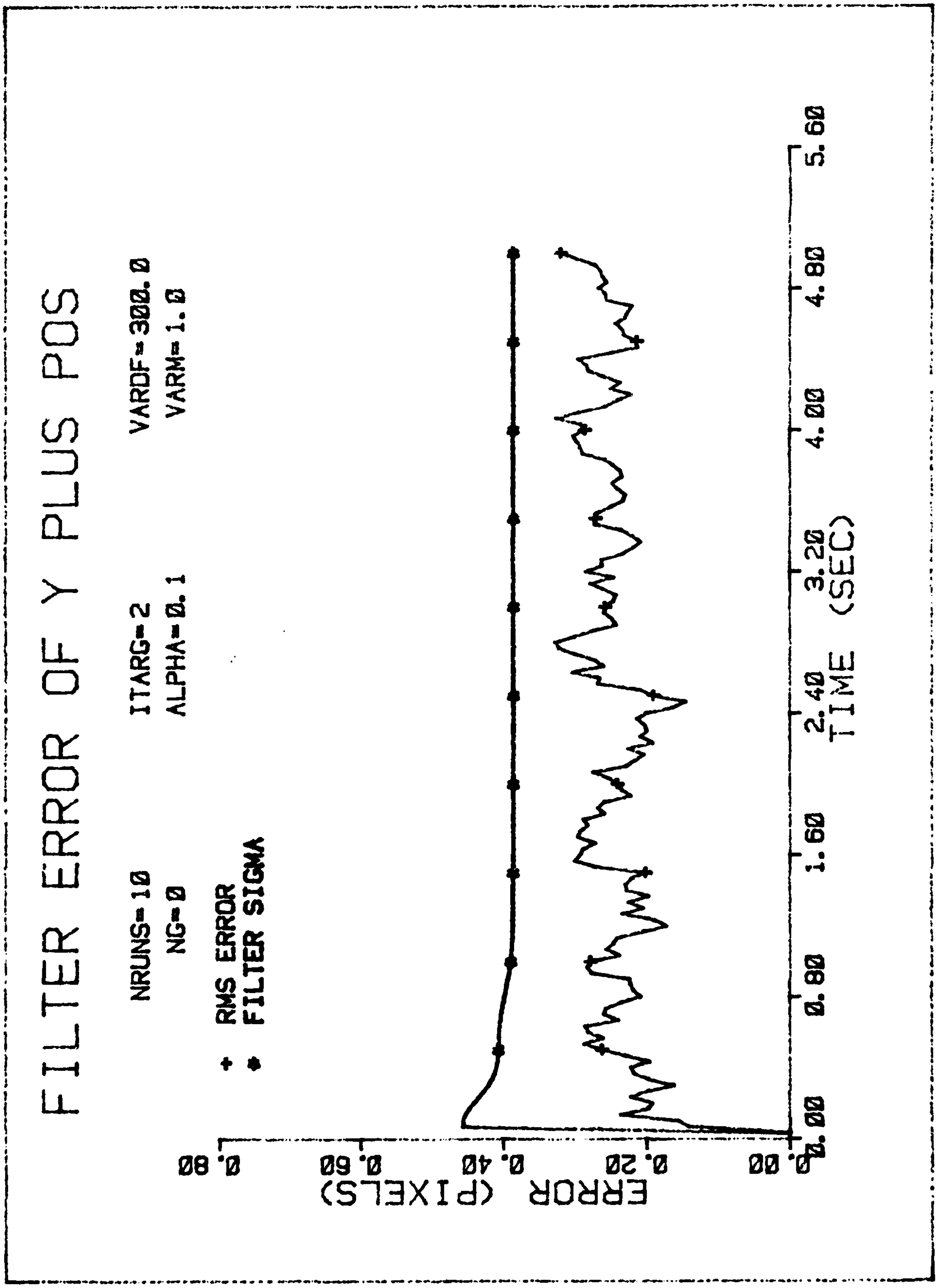


Figure D-10 Case 1 GM Performance Plot

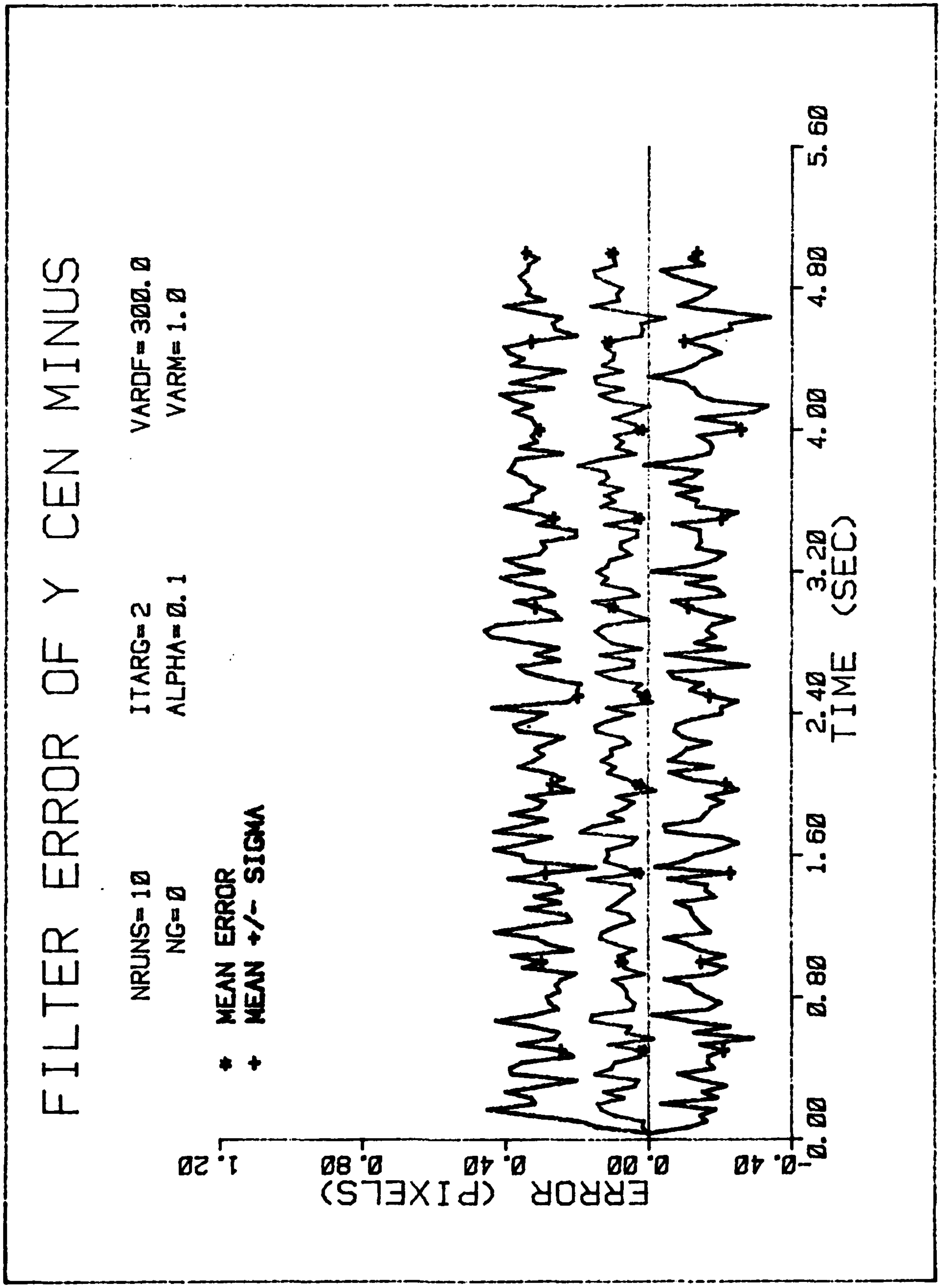


Figure D-11 Case 1 GM Performance Plot

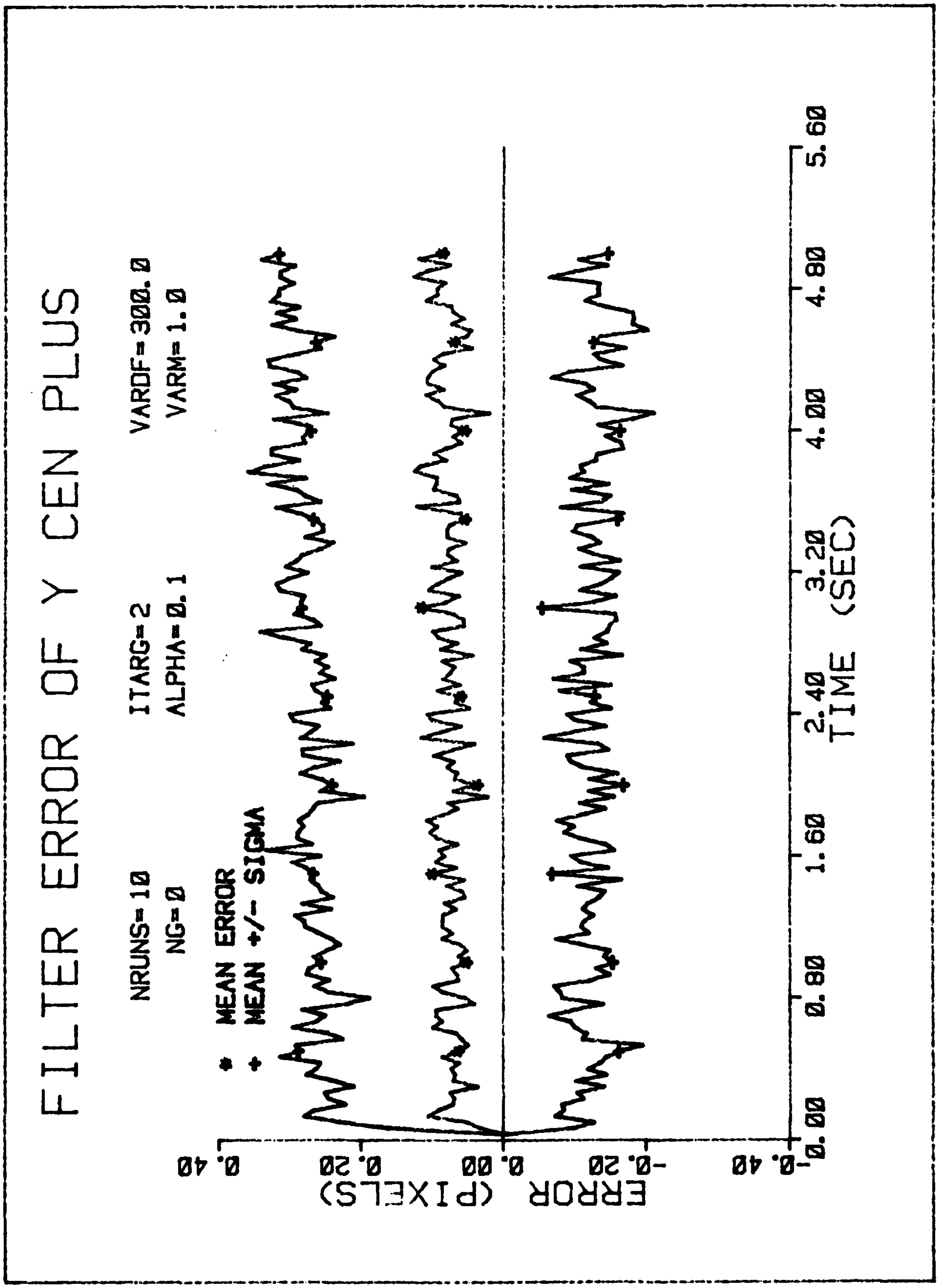


Figure D-12 Case 1 GM Performance Plot

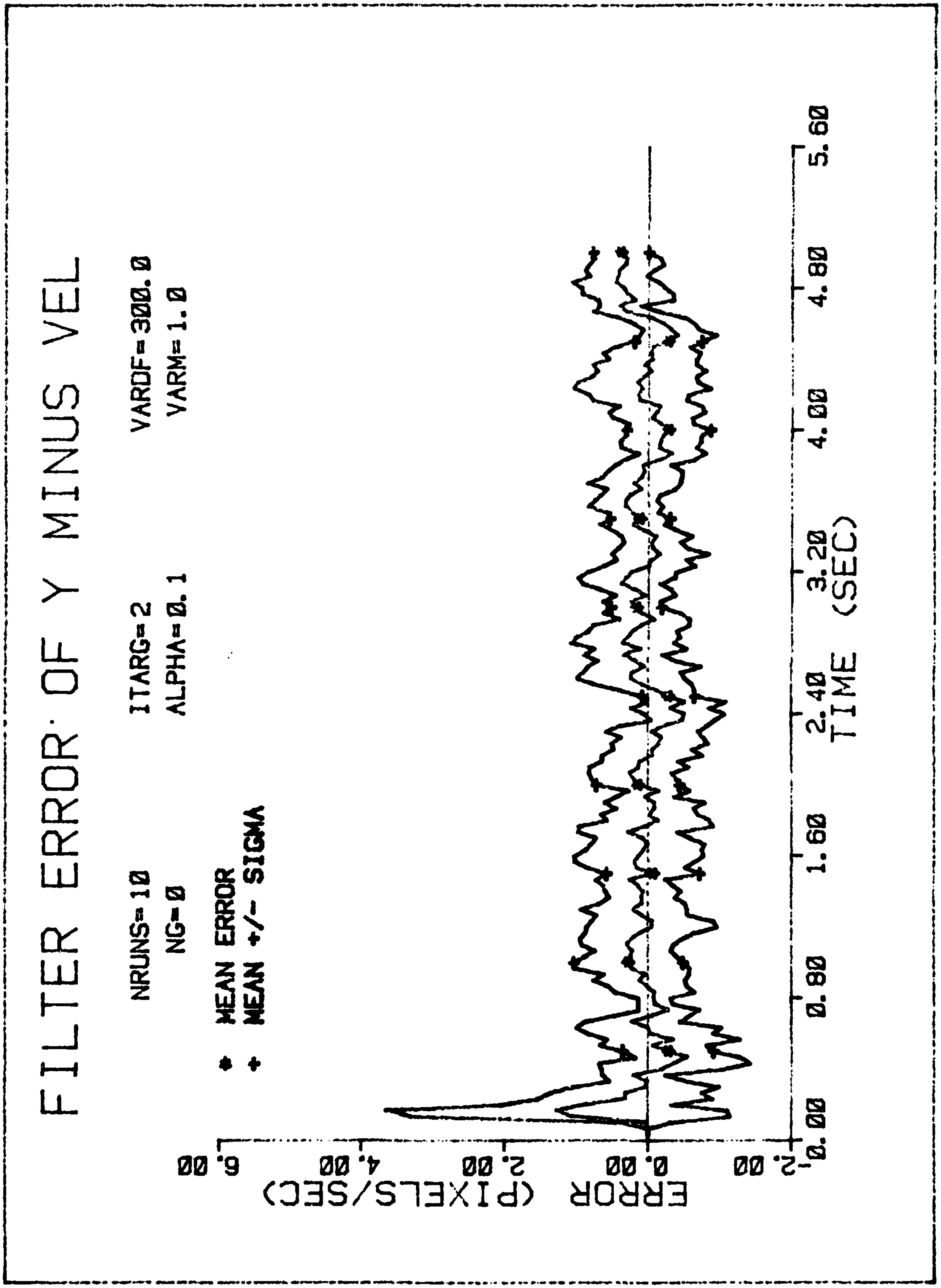


Figure D-13 Case 1 GM Performance Plot

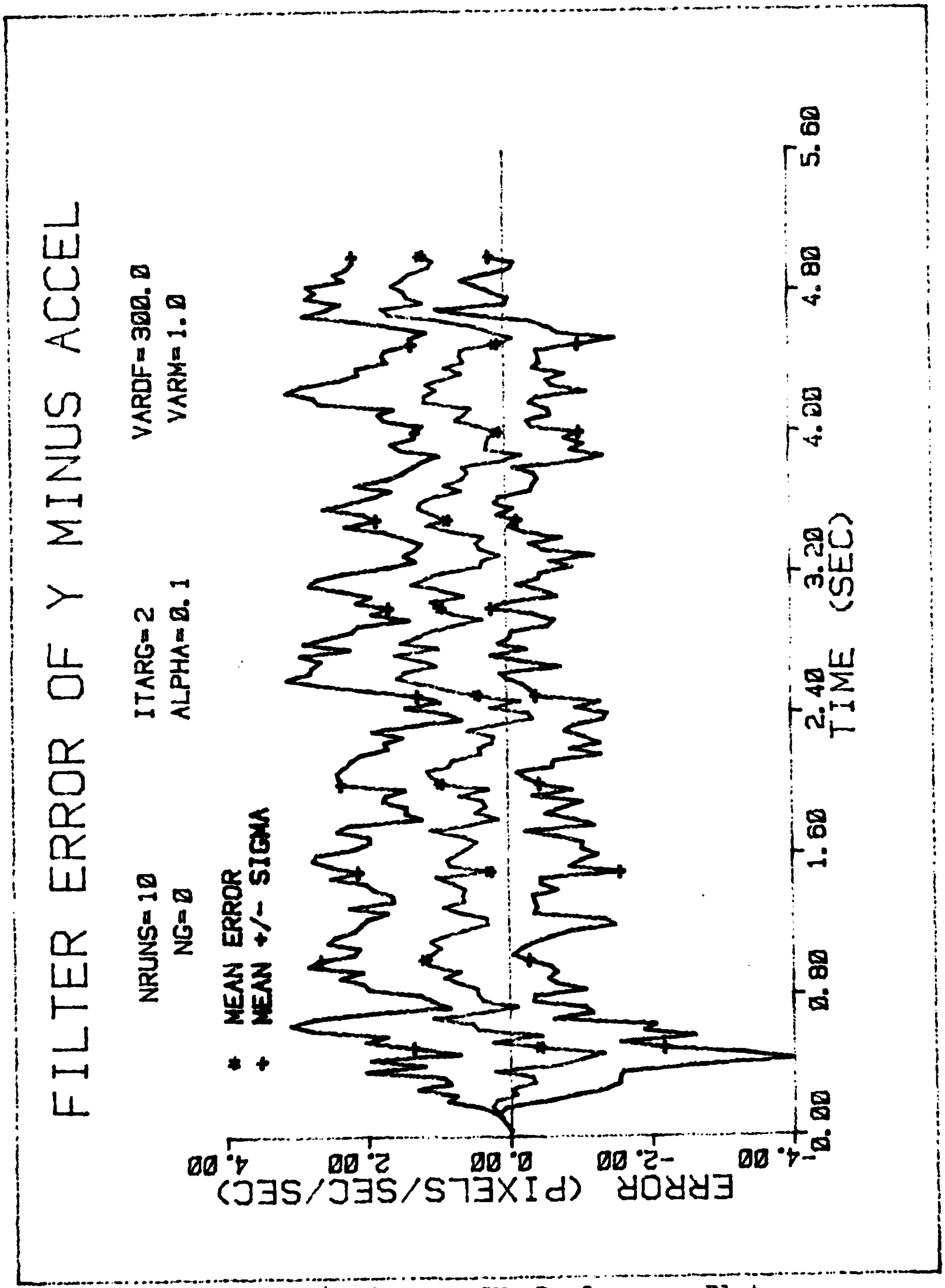


Figure D-14 Case 1 GM Performance Plot

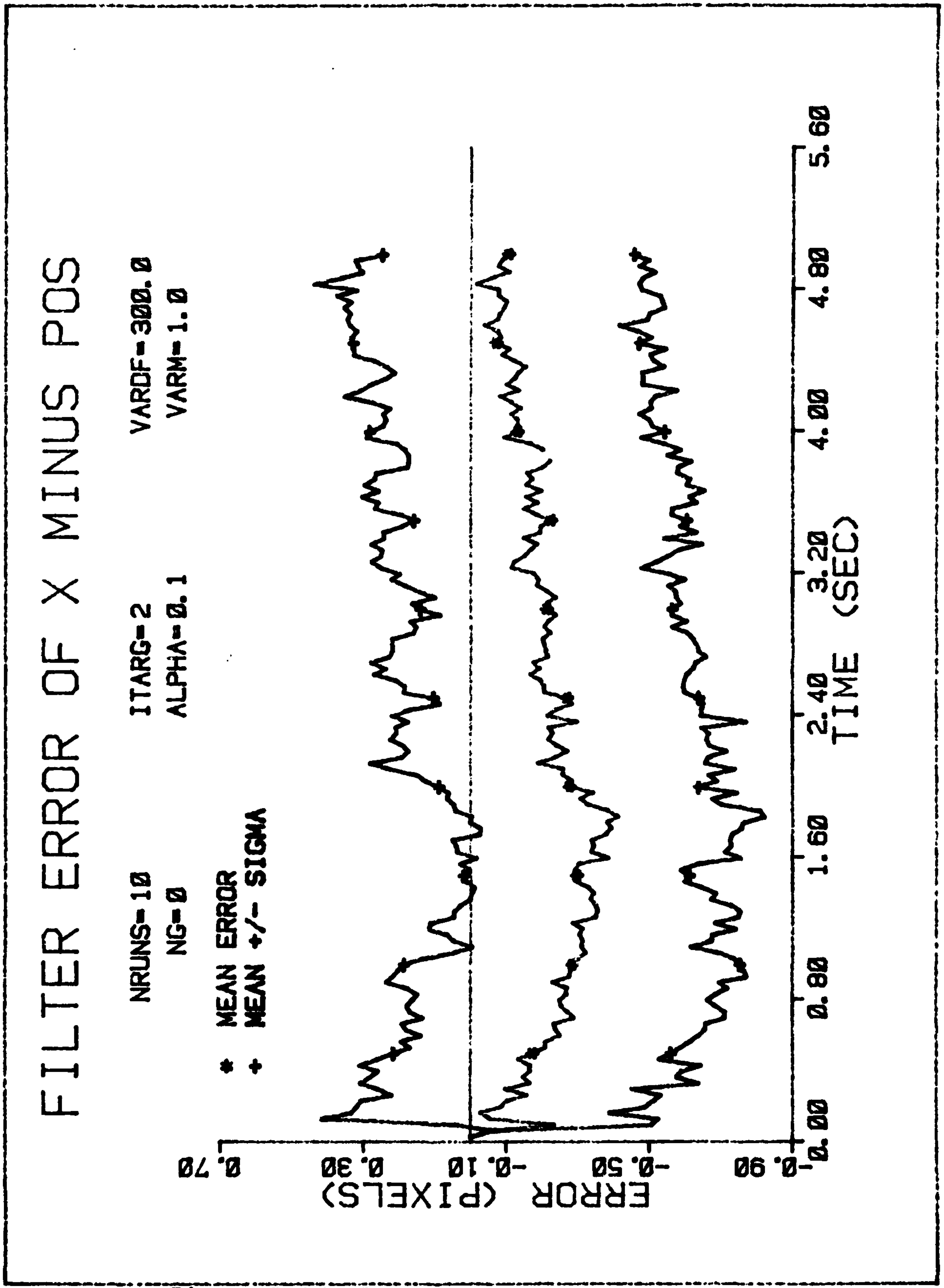


Figure D-15 Case 2 GM Performance Plot

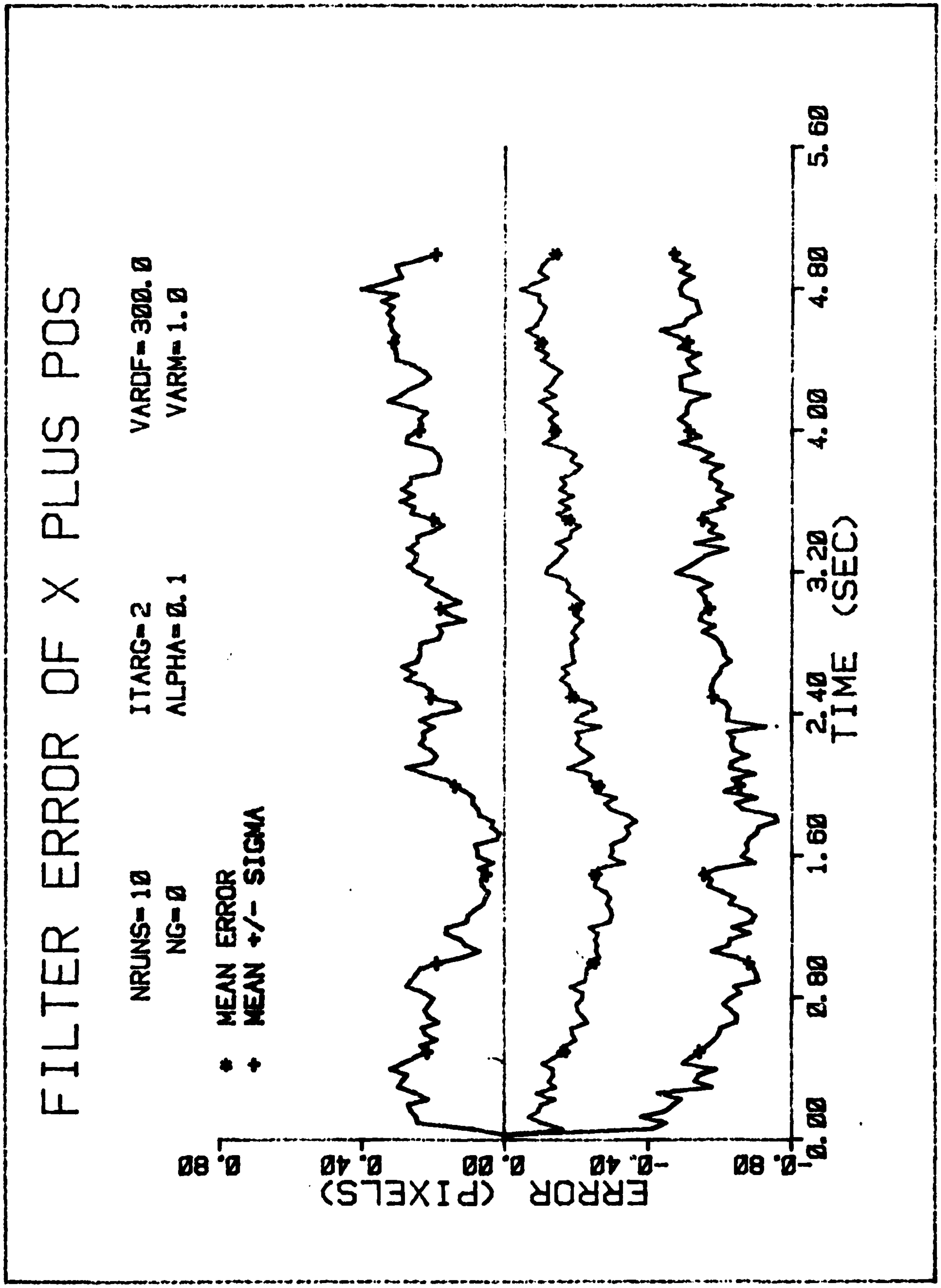


Figure D-16 Case 2 GM Performance Plot

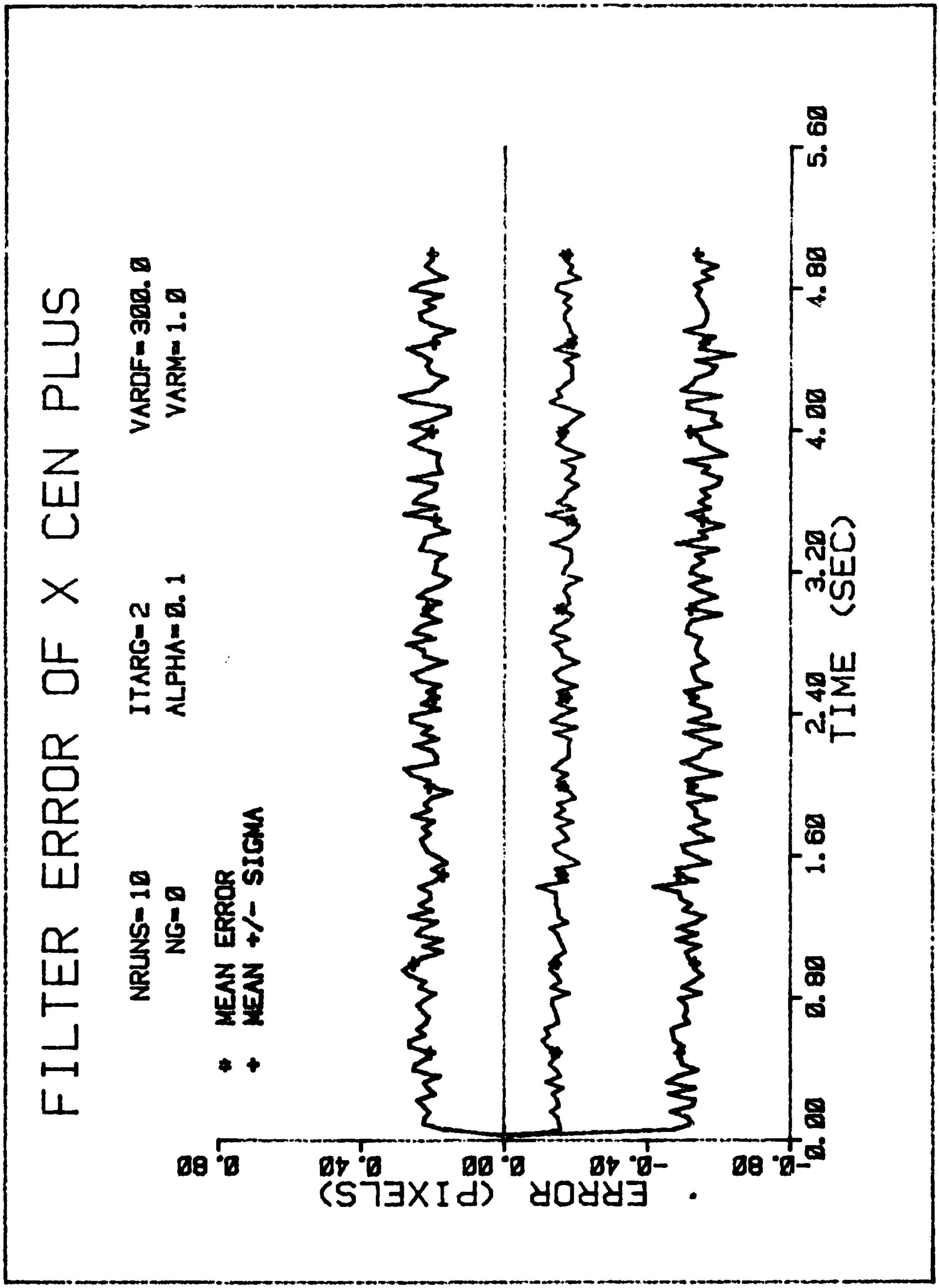


Figure D-17 Case 2 GM Performance Plot

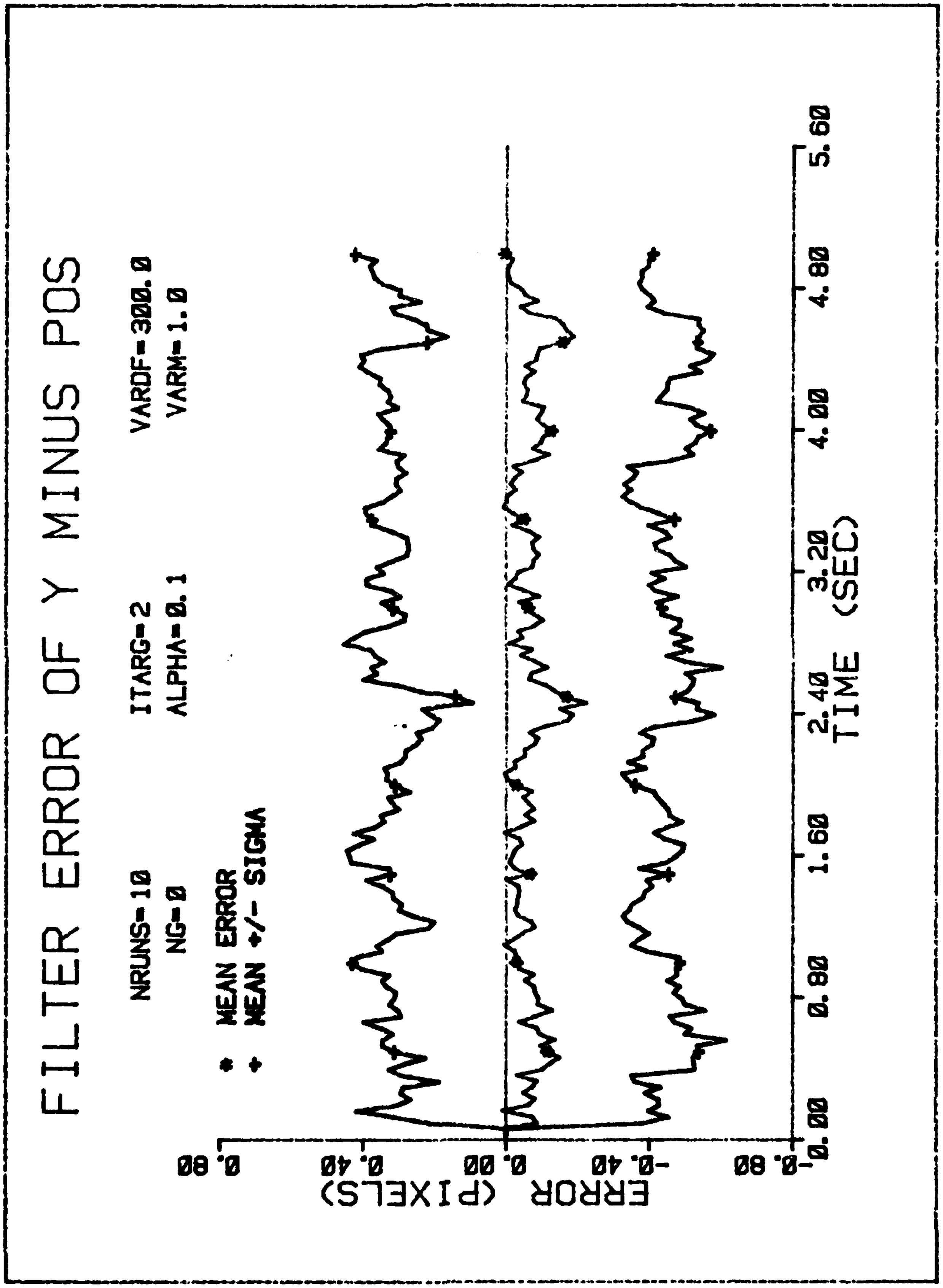


Figure D-18 Case 2 GM Performance Plot

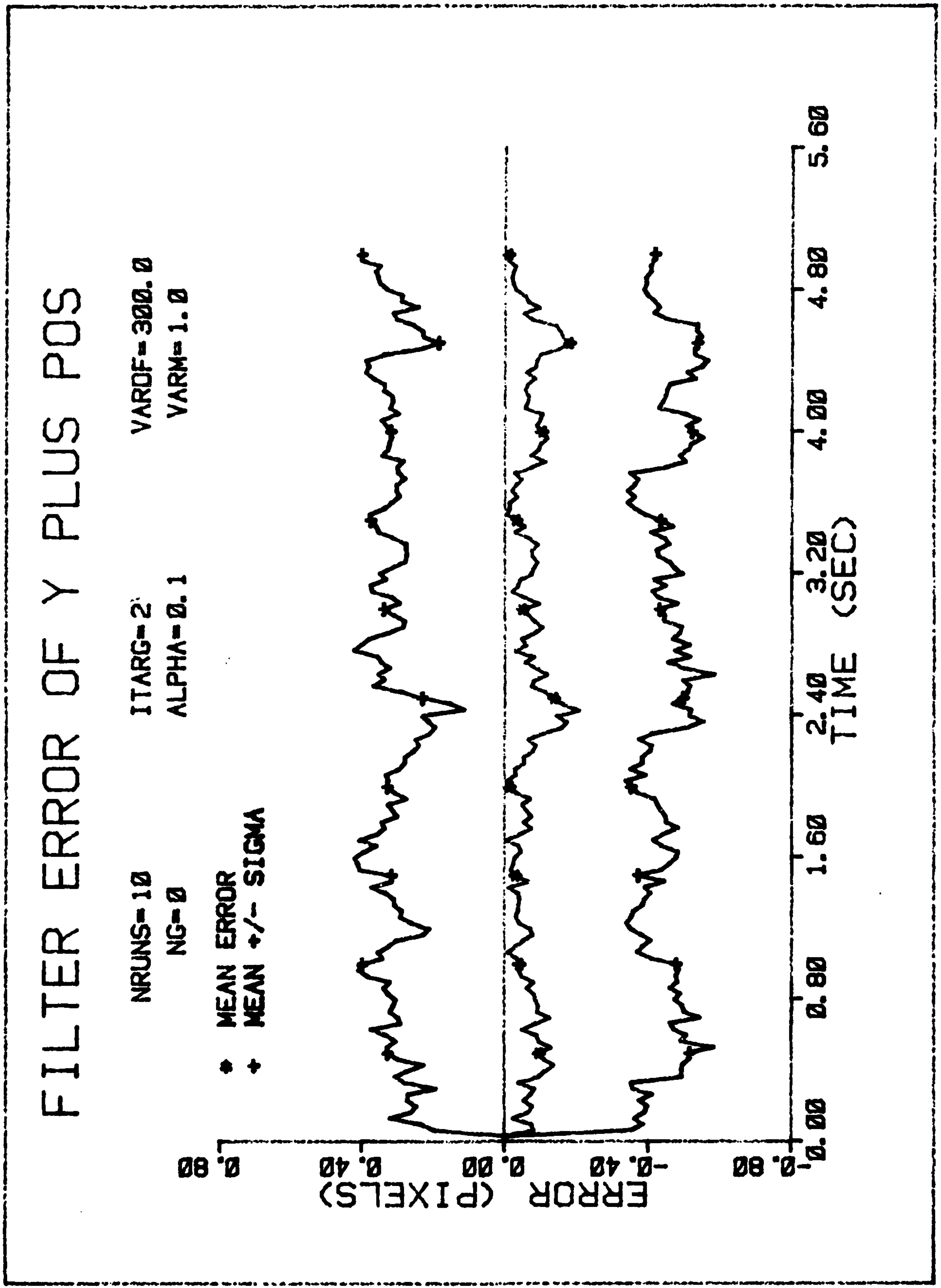


Figure D-19 Case 2 GM Performance Plot

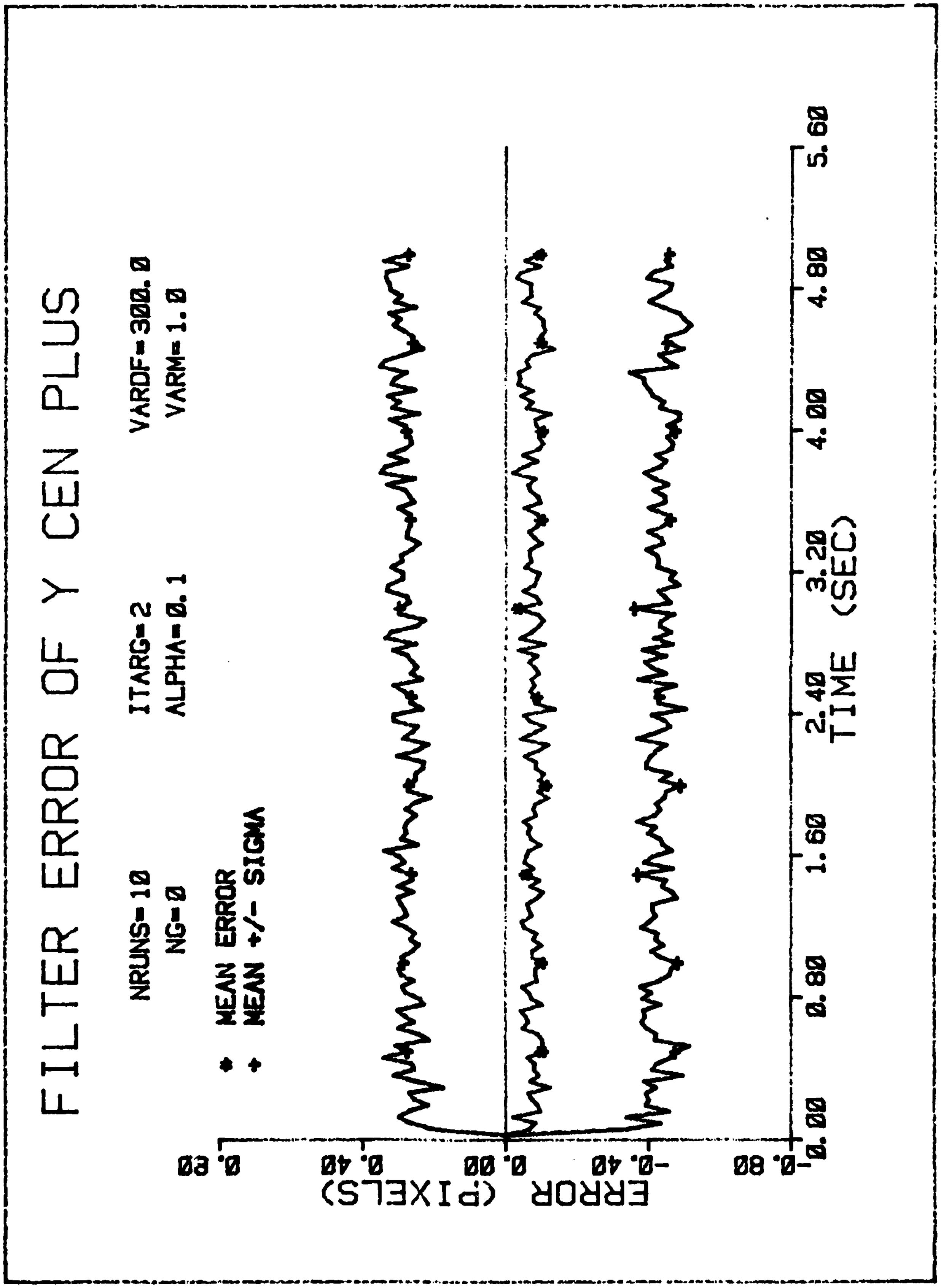


Figure D-20 Case 2 GM Performance Plot

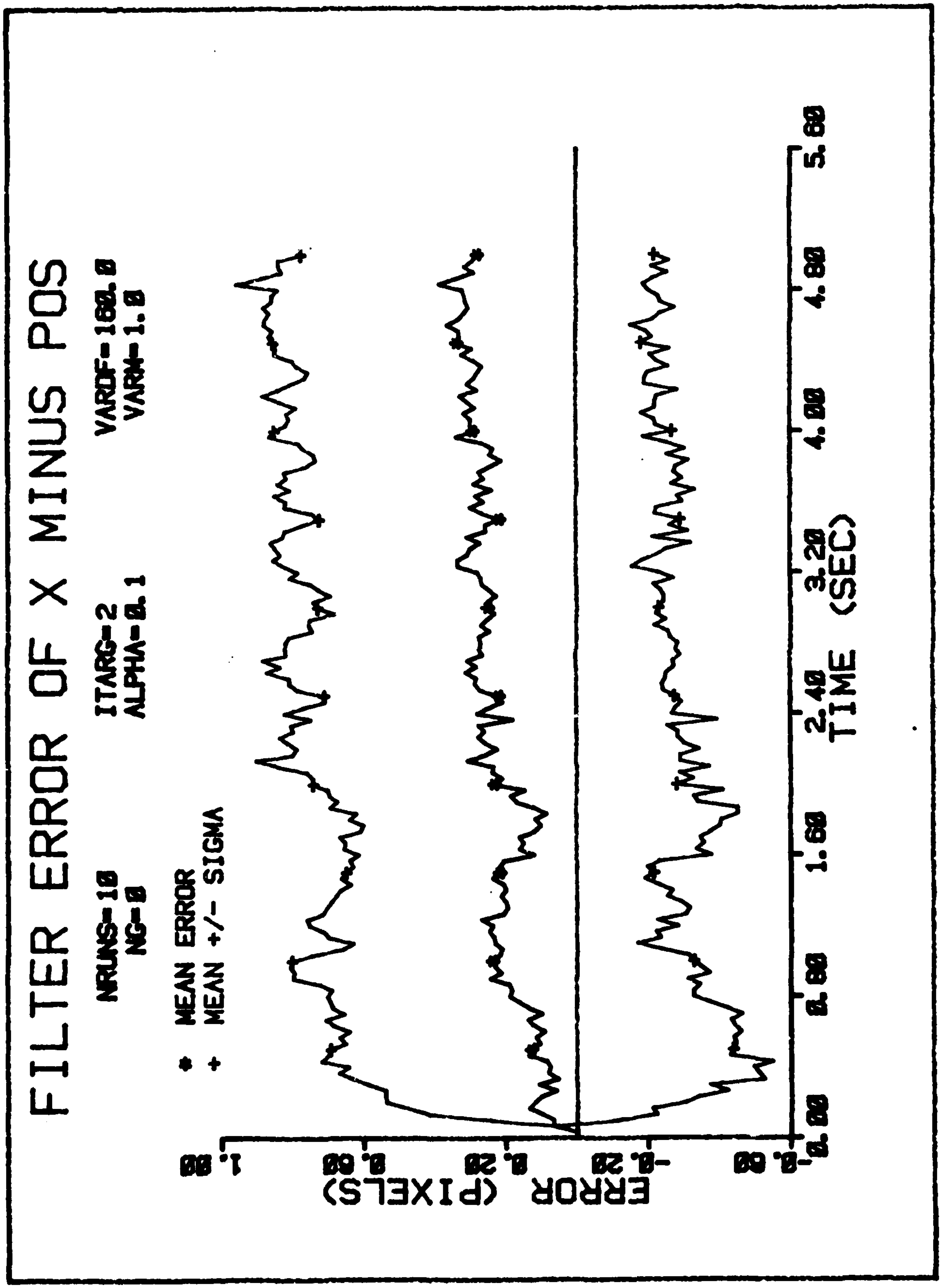


Figure D-21 Case 3 GM Performance Plot

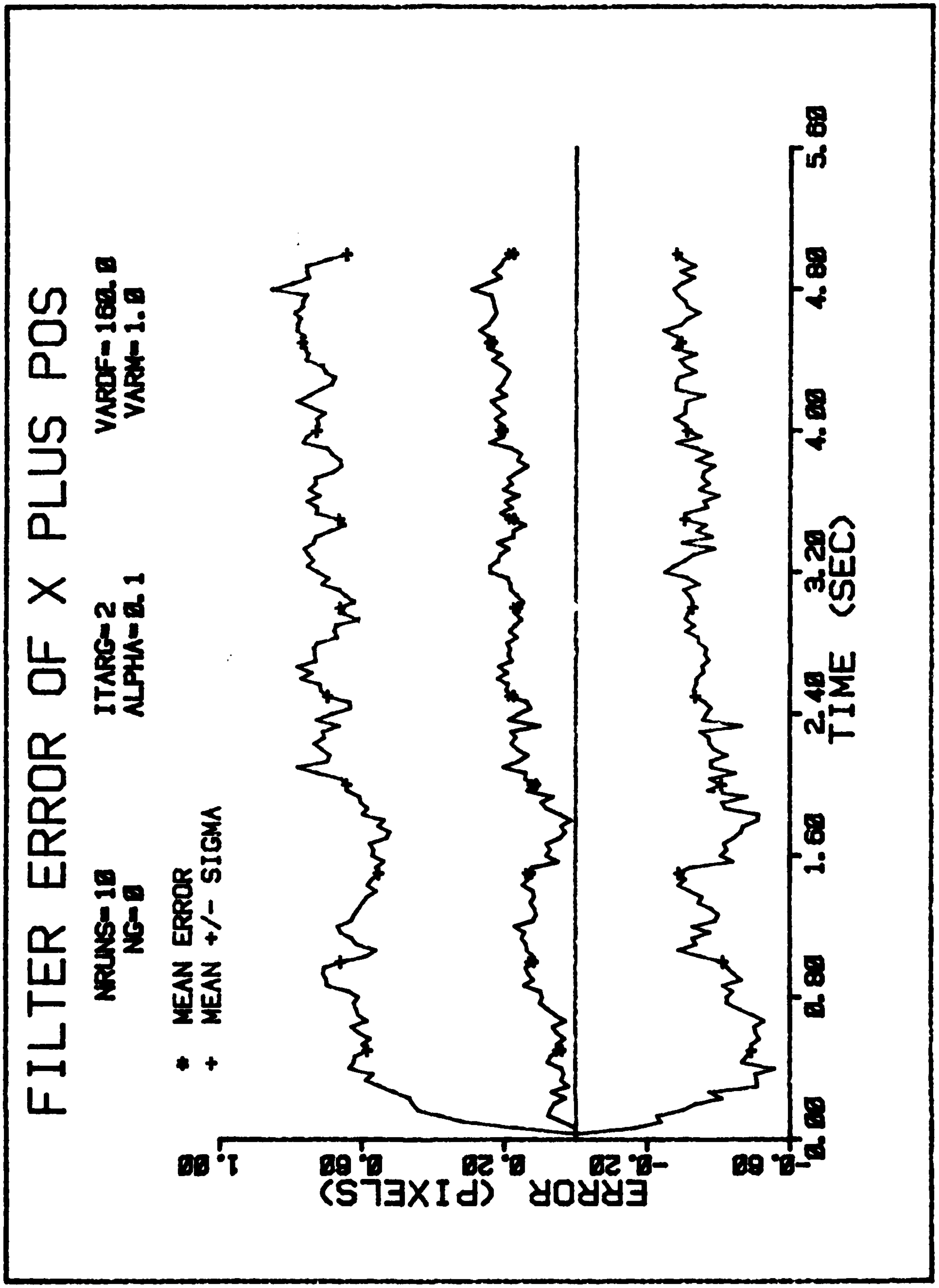


Figure D-22 Case 3 GM Performance Plot

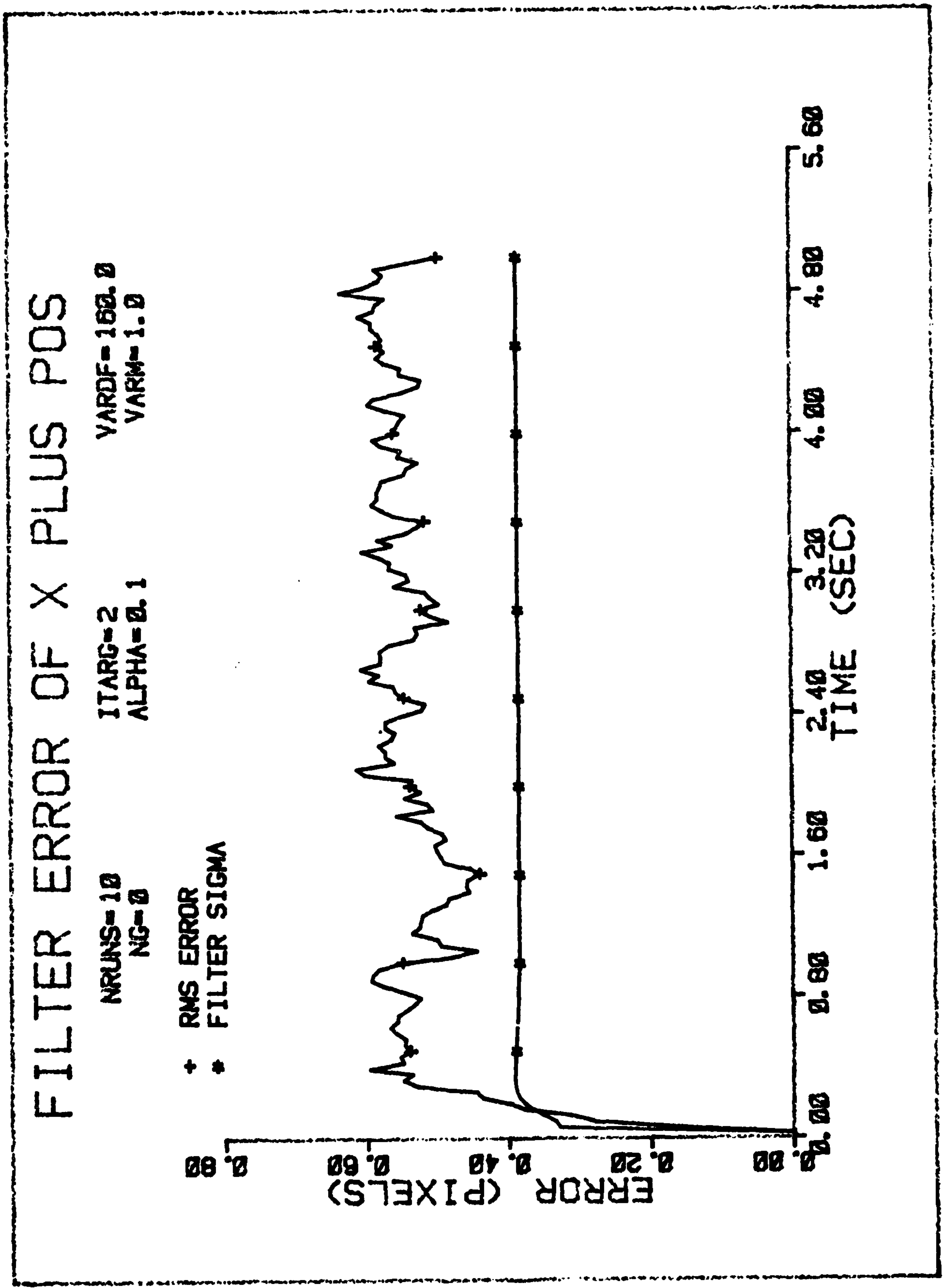


Figure D-23 Case 3 GM Performance Plot

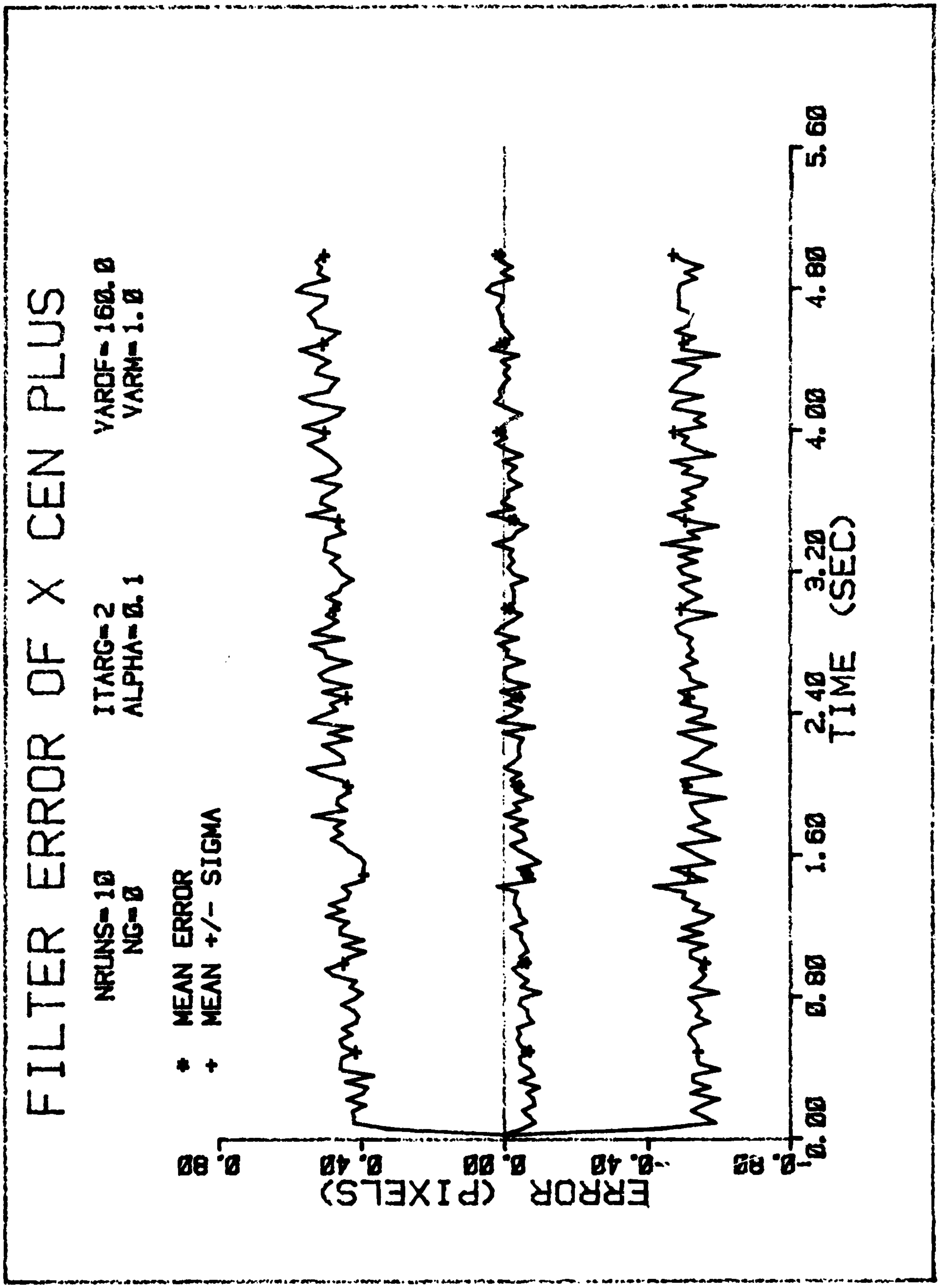
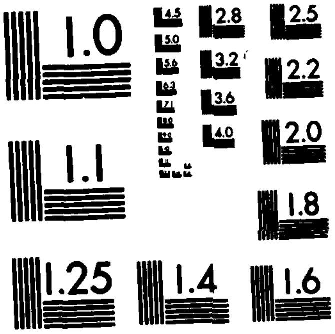


Figure D-24 Case 3 GM Performance Plot

AD-A124 781 ENHANCED IMAGE TRACKING: ANALYSIS OF TWO ACCELERATION 3/6
MODELS IN TRACKING..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. M R KOZEMCHAK
UNCLASSIFIED DEC 82 AFIT/GEO/EE/82D-4 F/G 17/7 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

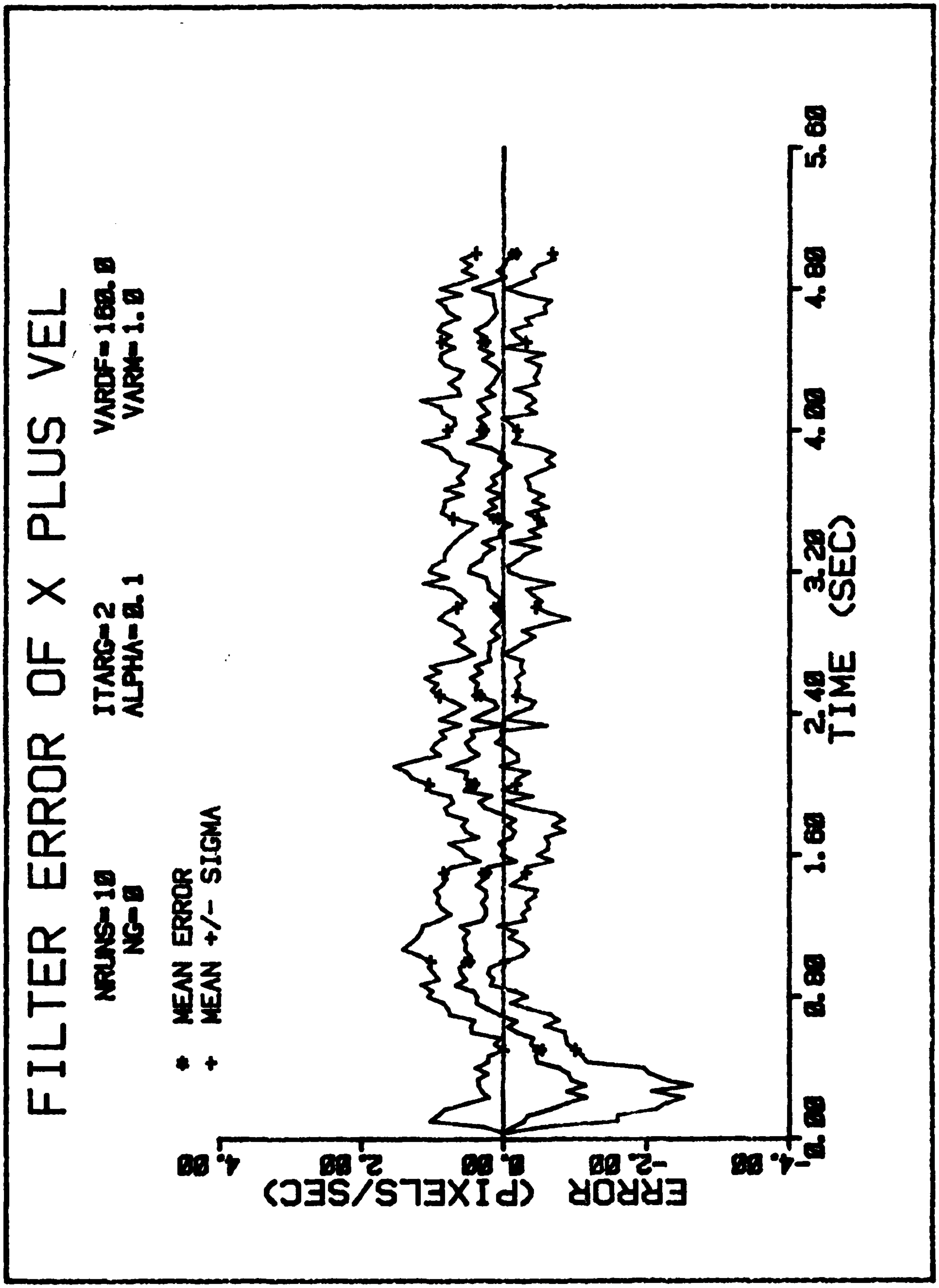


Figure D-25 Case 3 GM Performance Plot

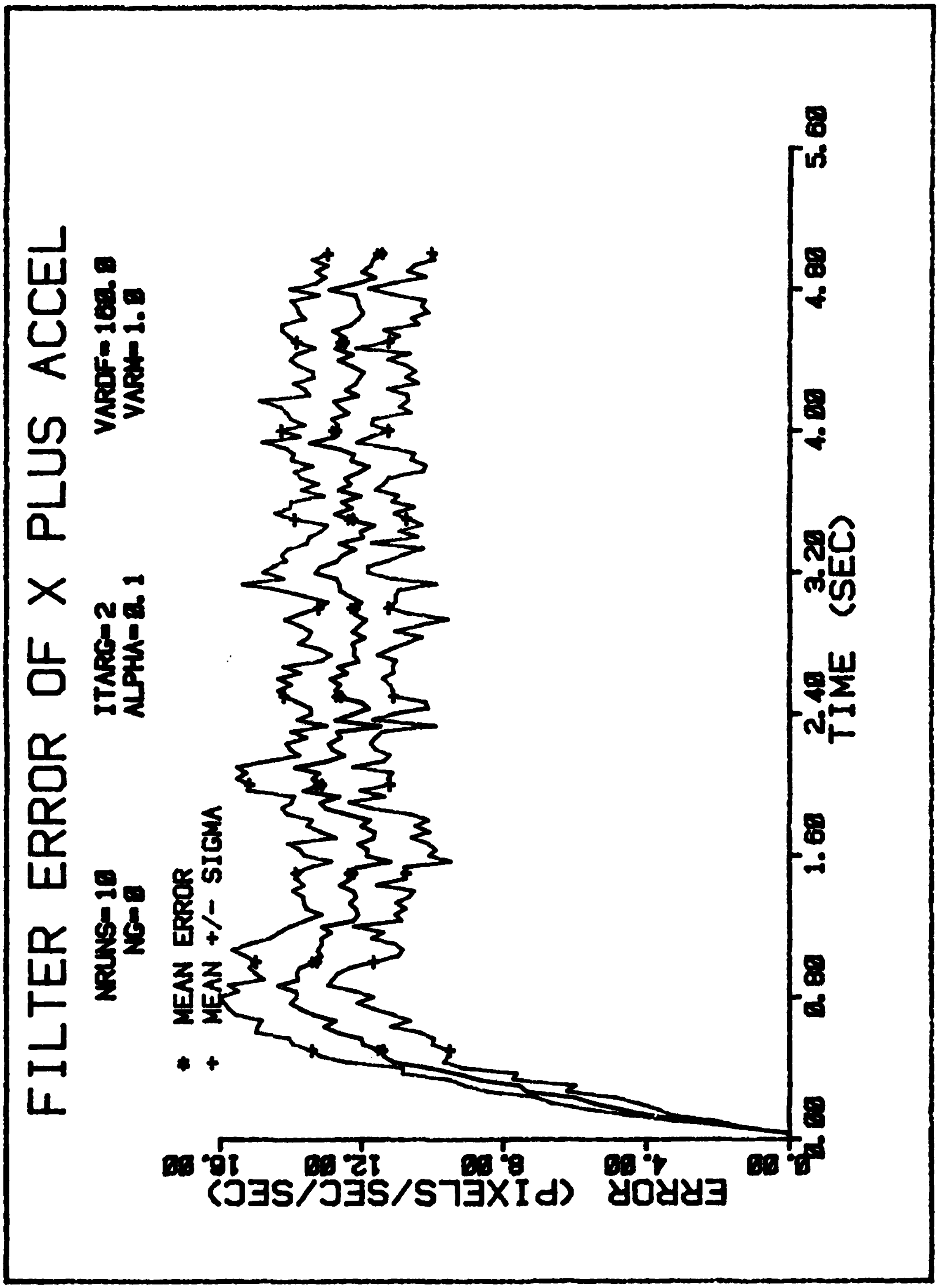


Figure D-26 Case 3 GM Performance Plot

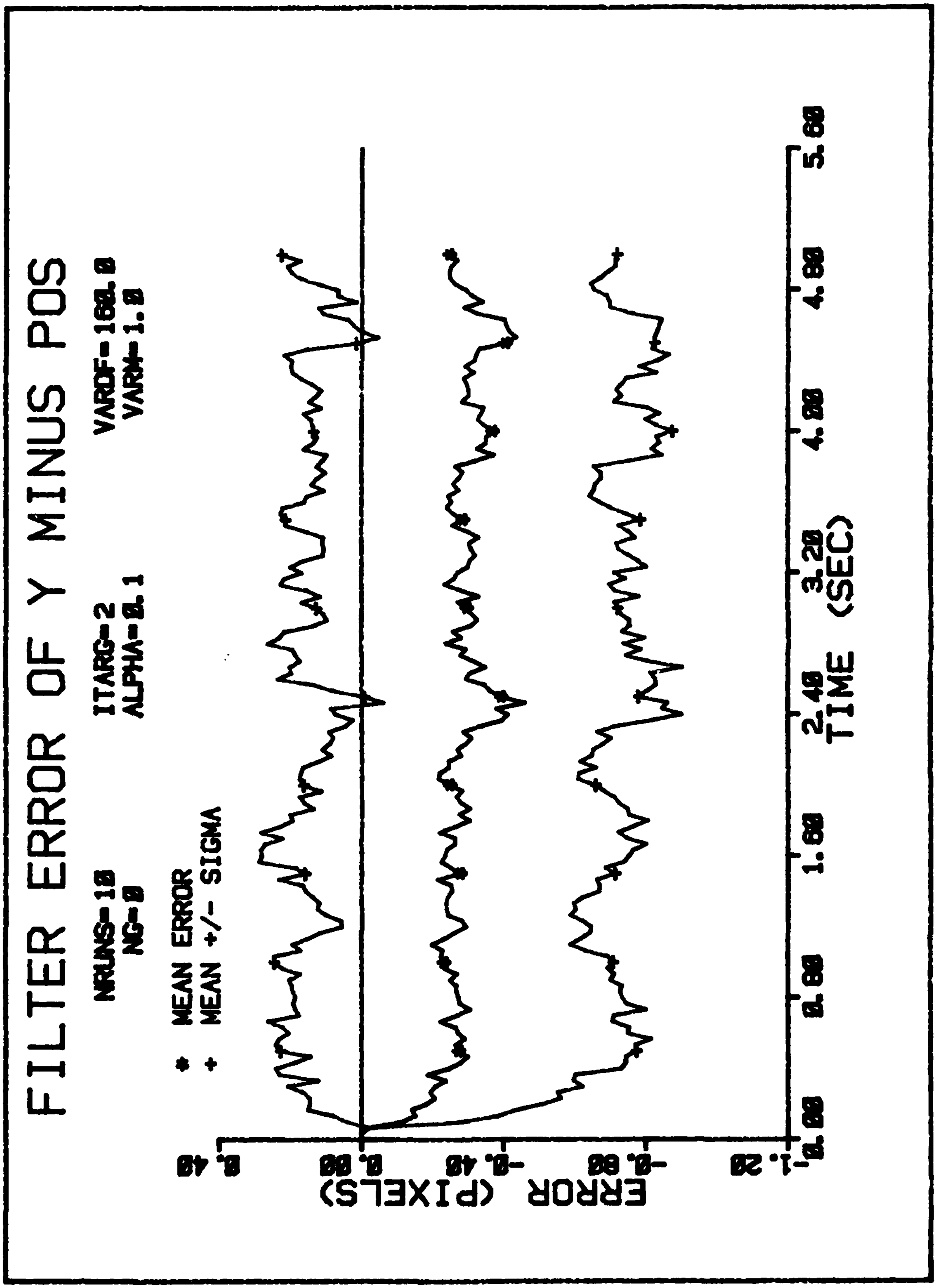


Figure D-27 Case 3 GM Performance Plot

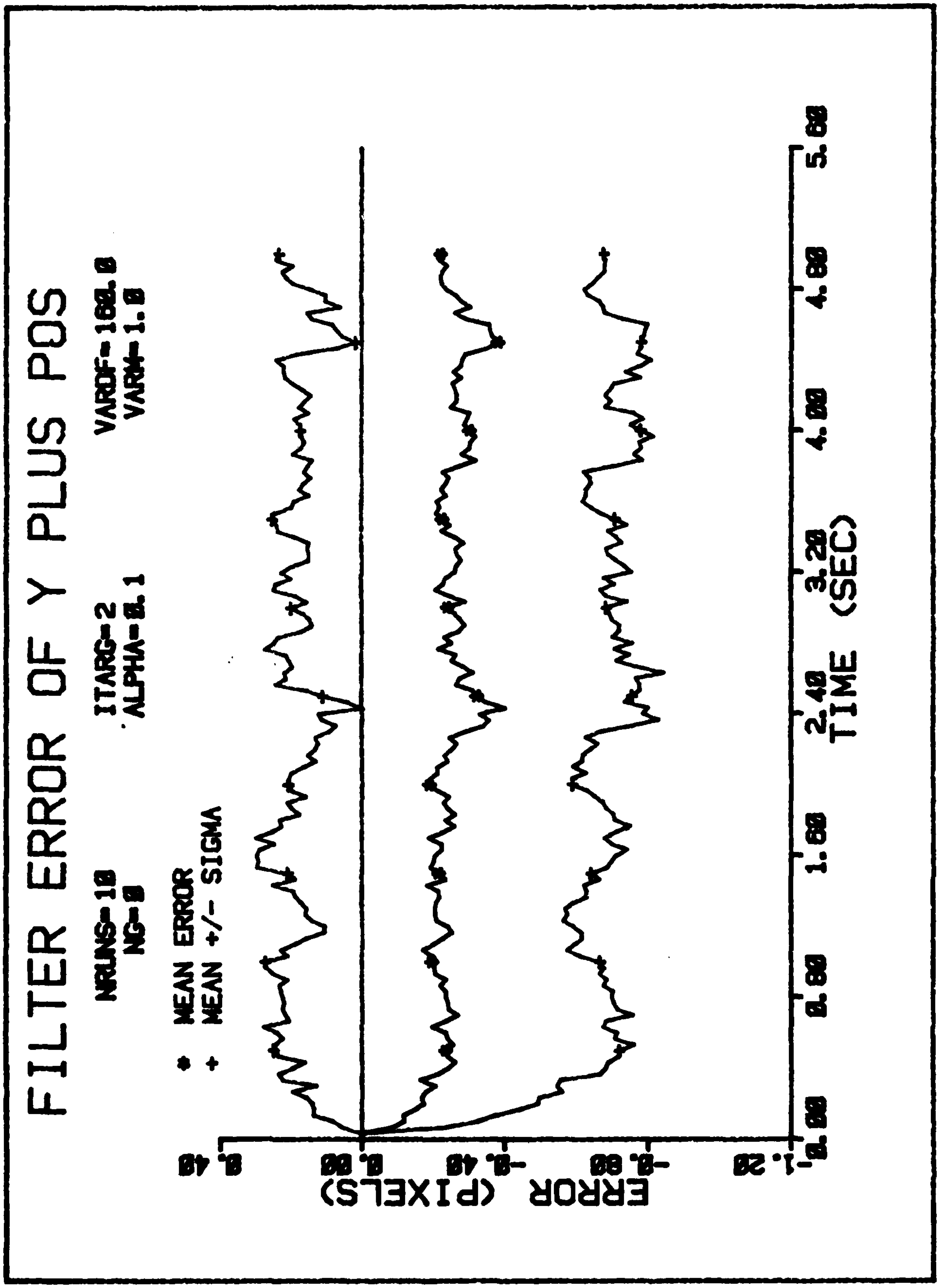


Figure D-28 Case 3 GM Performance Plot

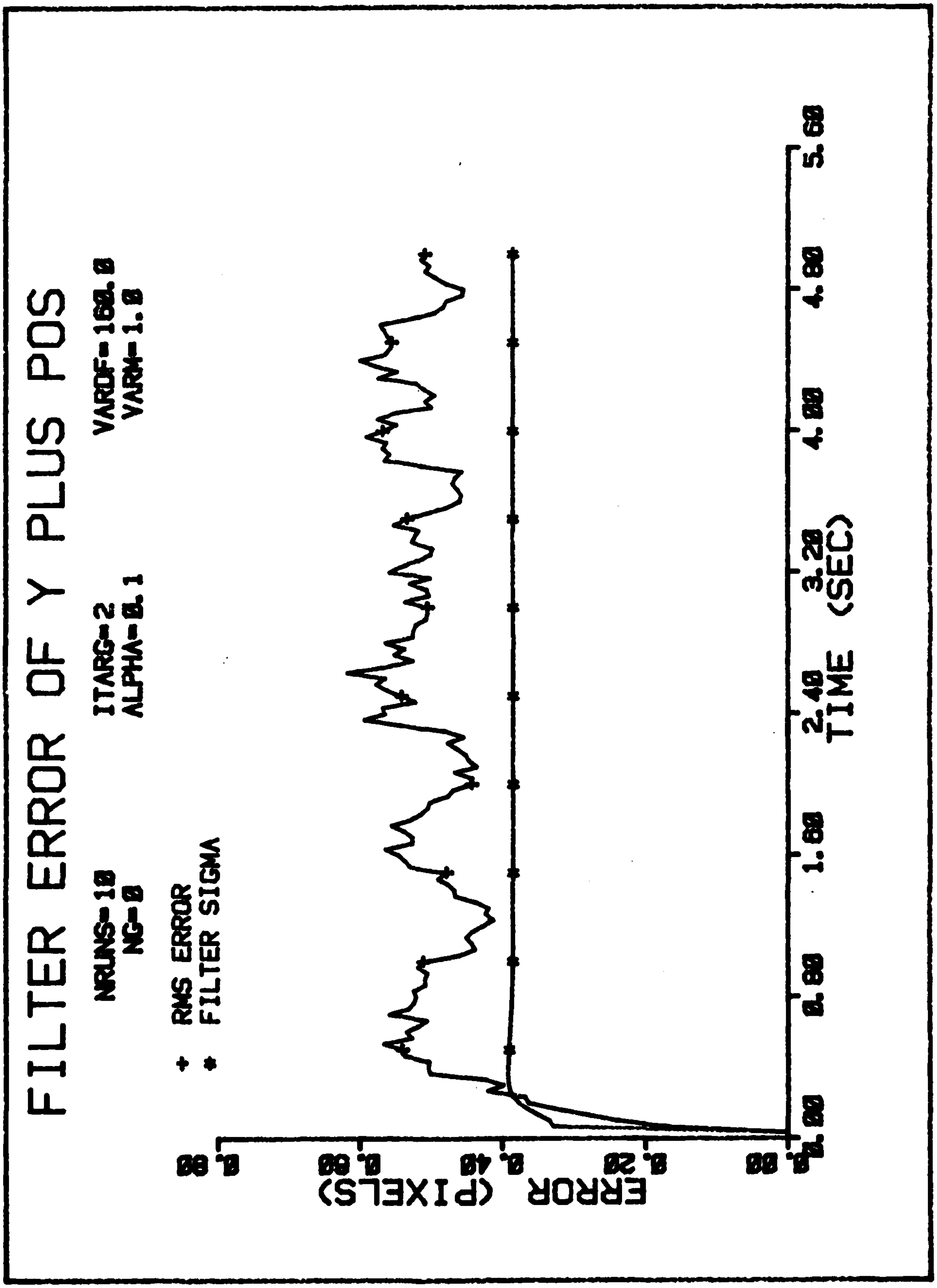


Figure D-29 Case 3 GM Performance Plot

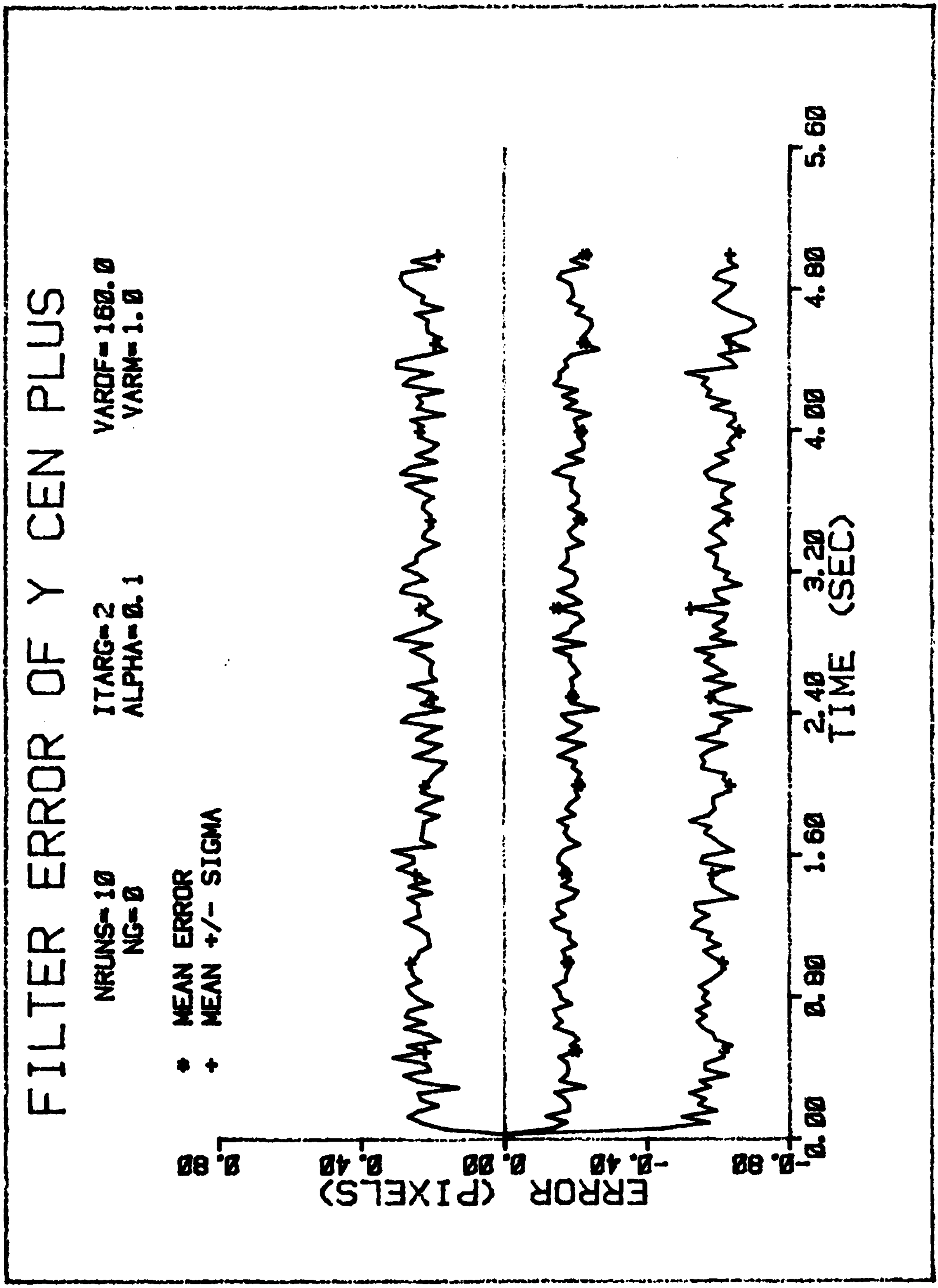


Figure D-30 Case 3 GM Performance Plot

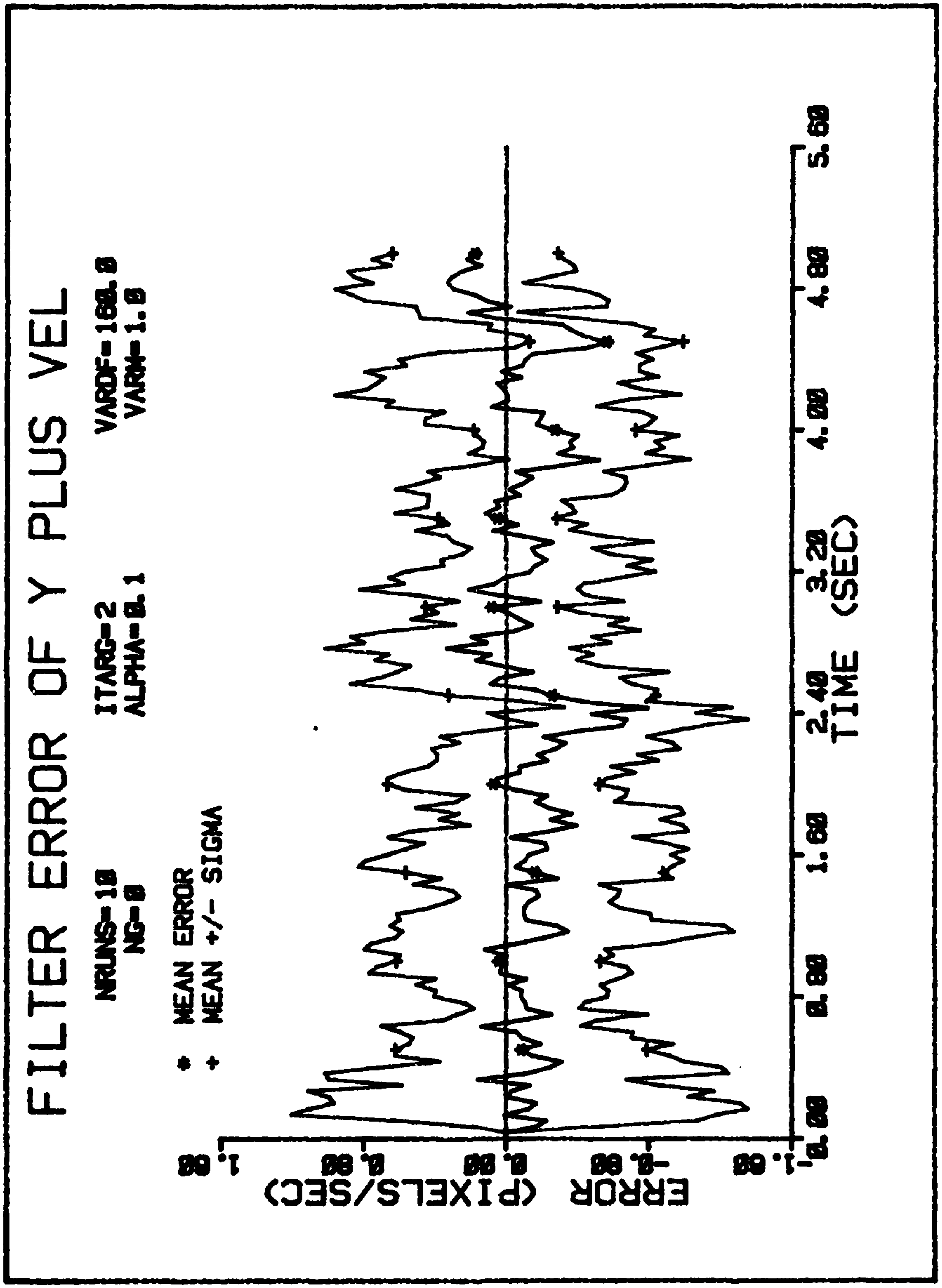


Figure D-31 Case 3 GM Performance Plot

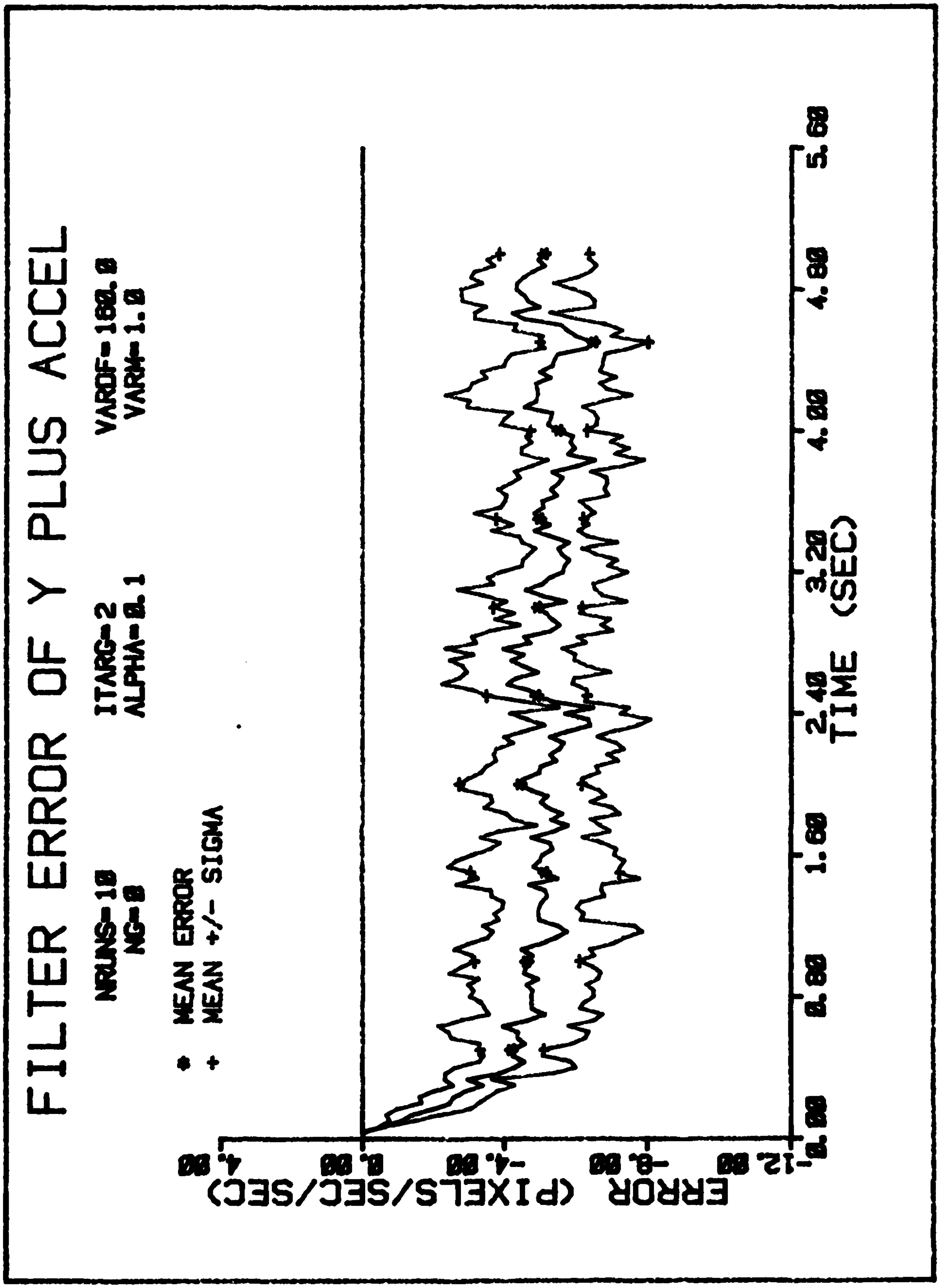


Figure D-32 Case 3 GM Performance Plot

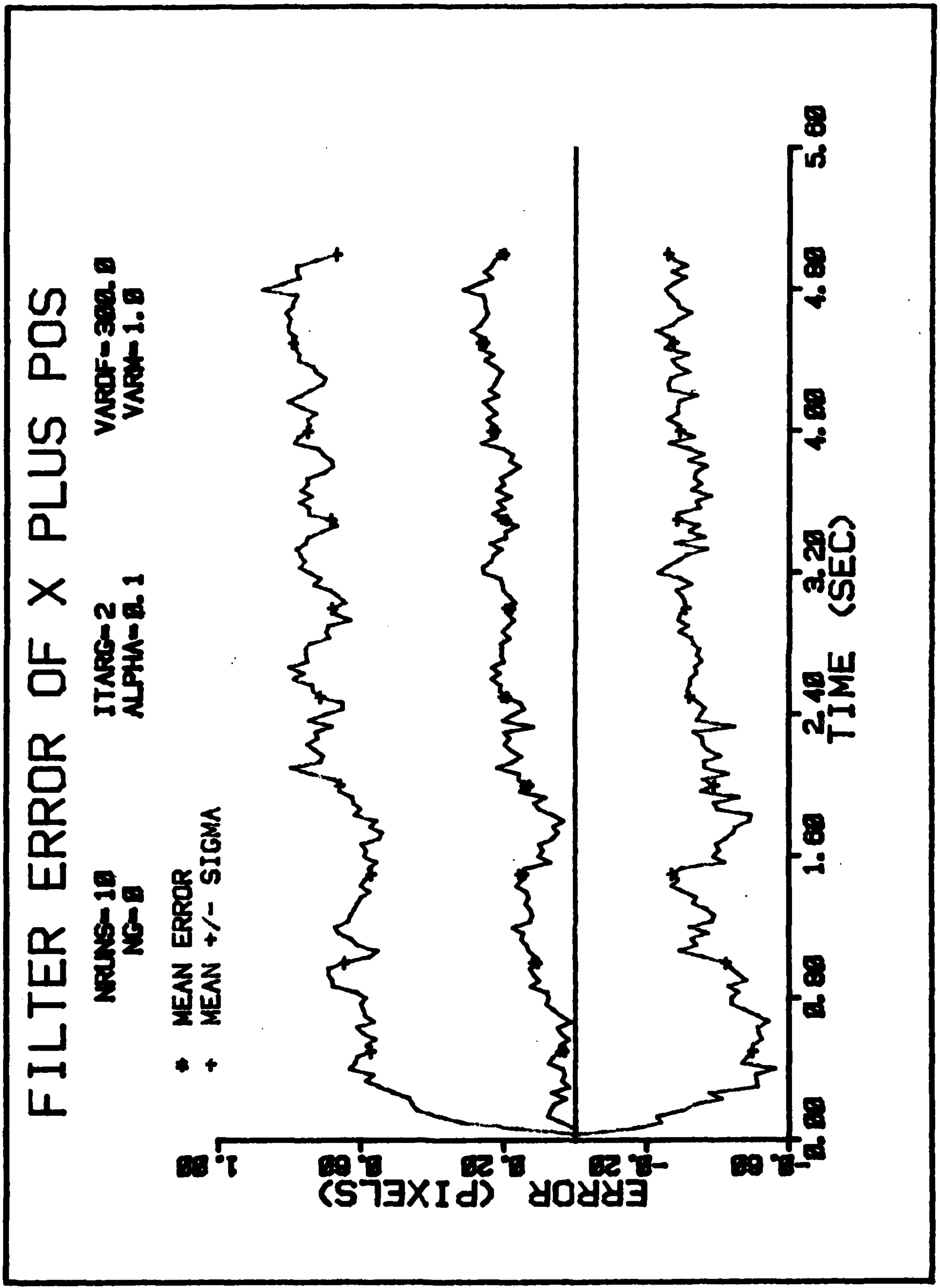


Figure D-33 Case 4 GM Performance Plot

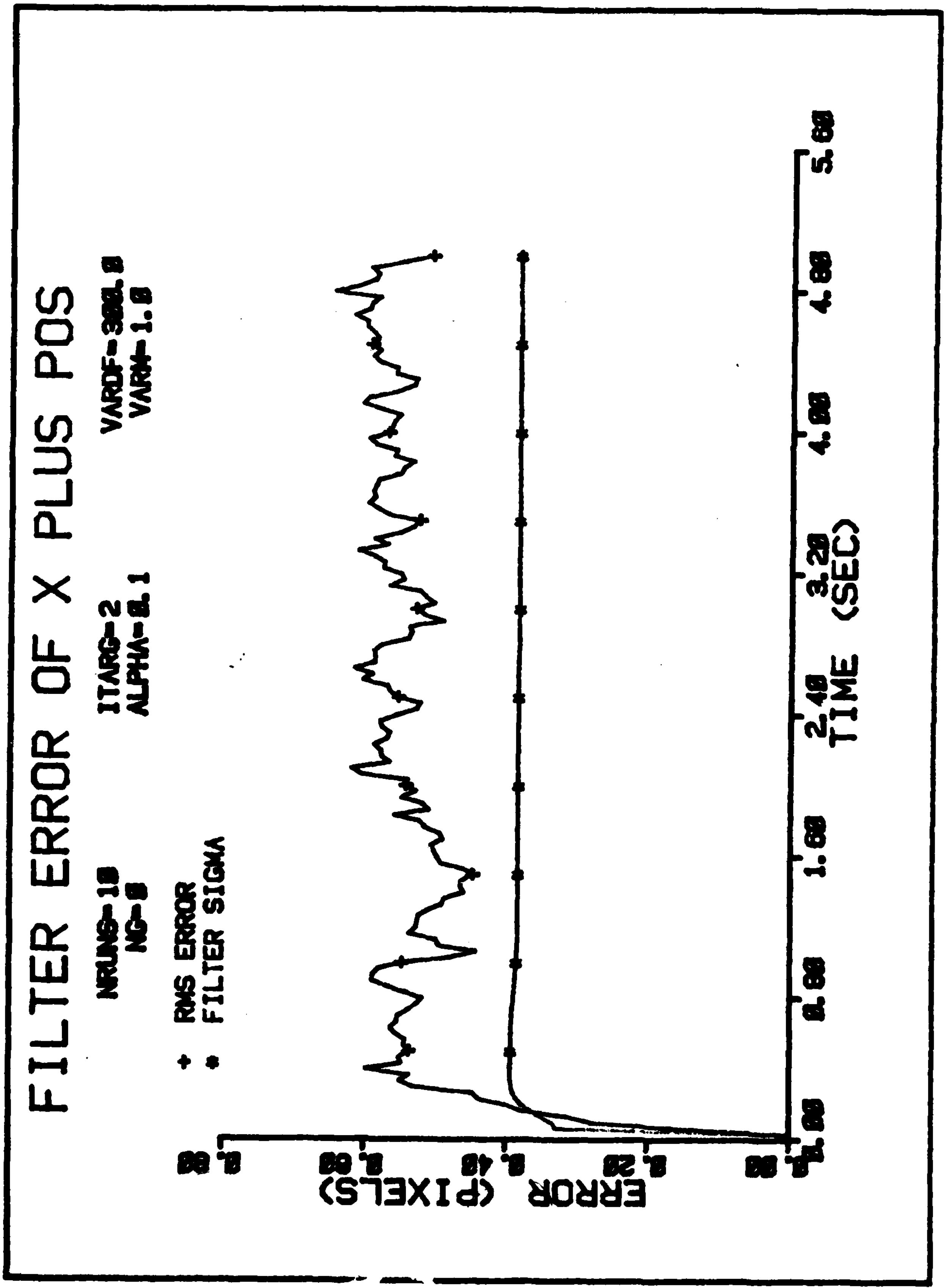


Figure D-34 Case 4 GM Performance Plot

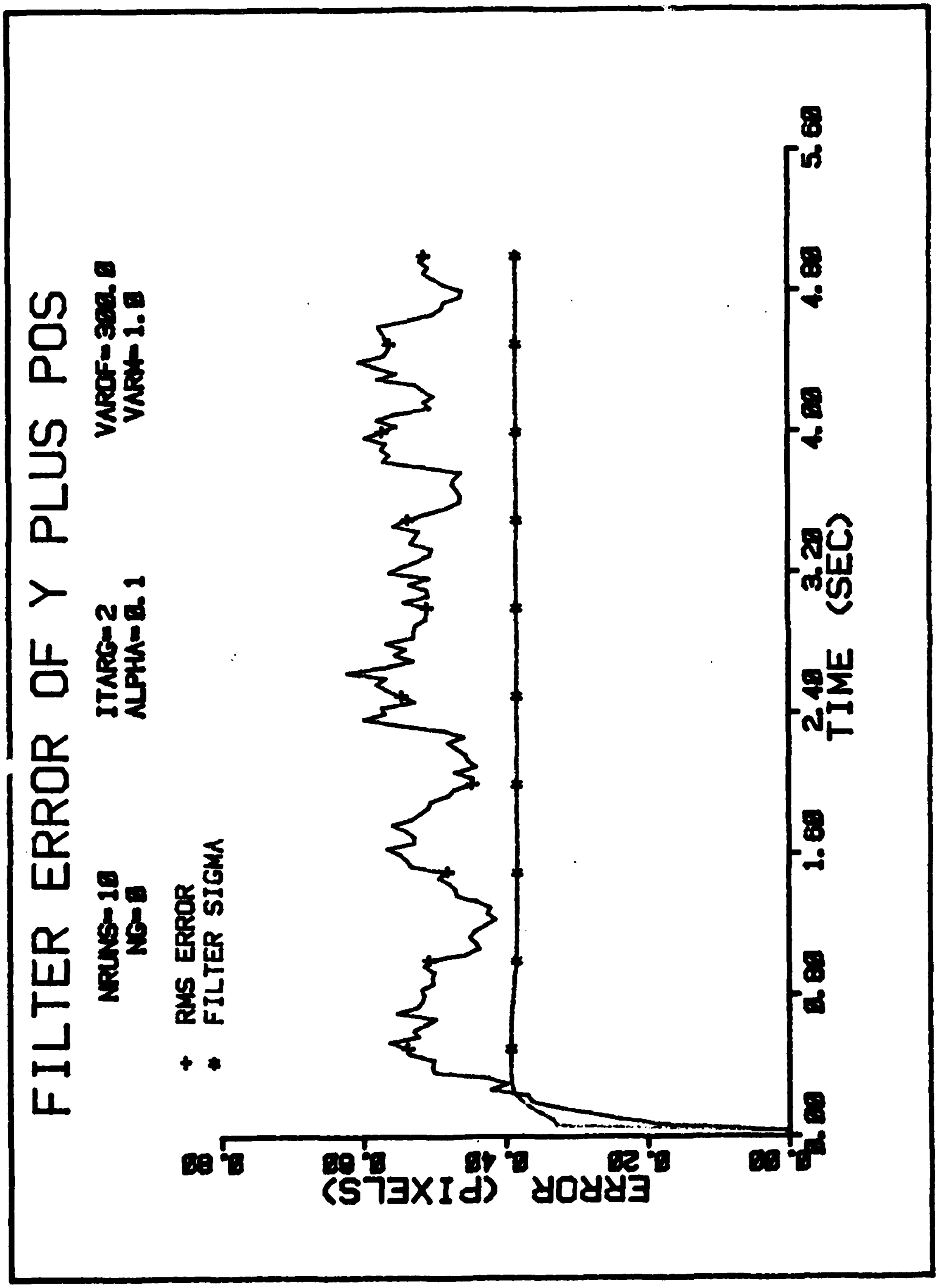


Figure D-35 Case 4 GM Performance Plot

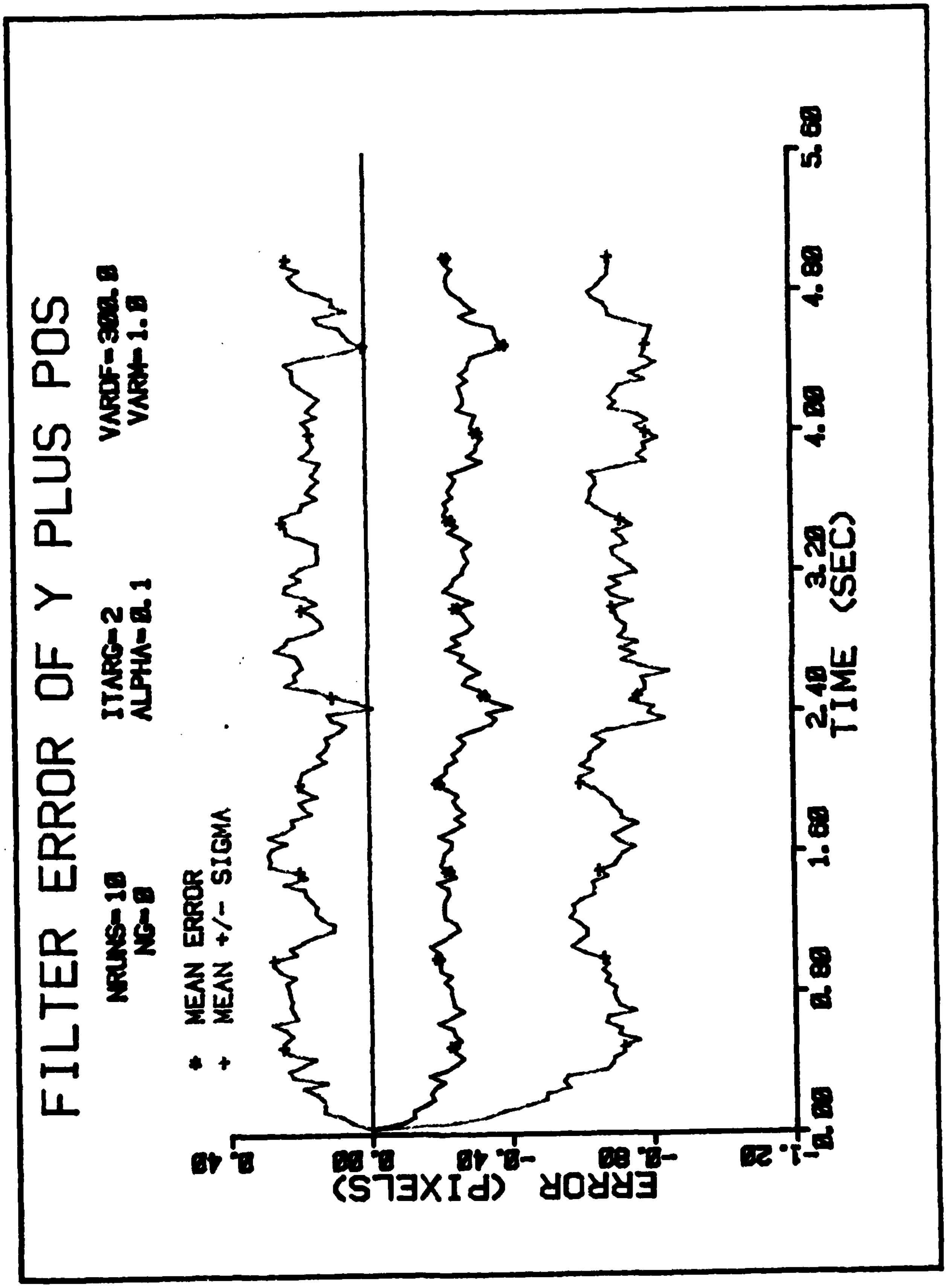


Figure D-36 Case 4 GM Performance Plot

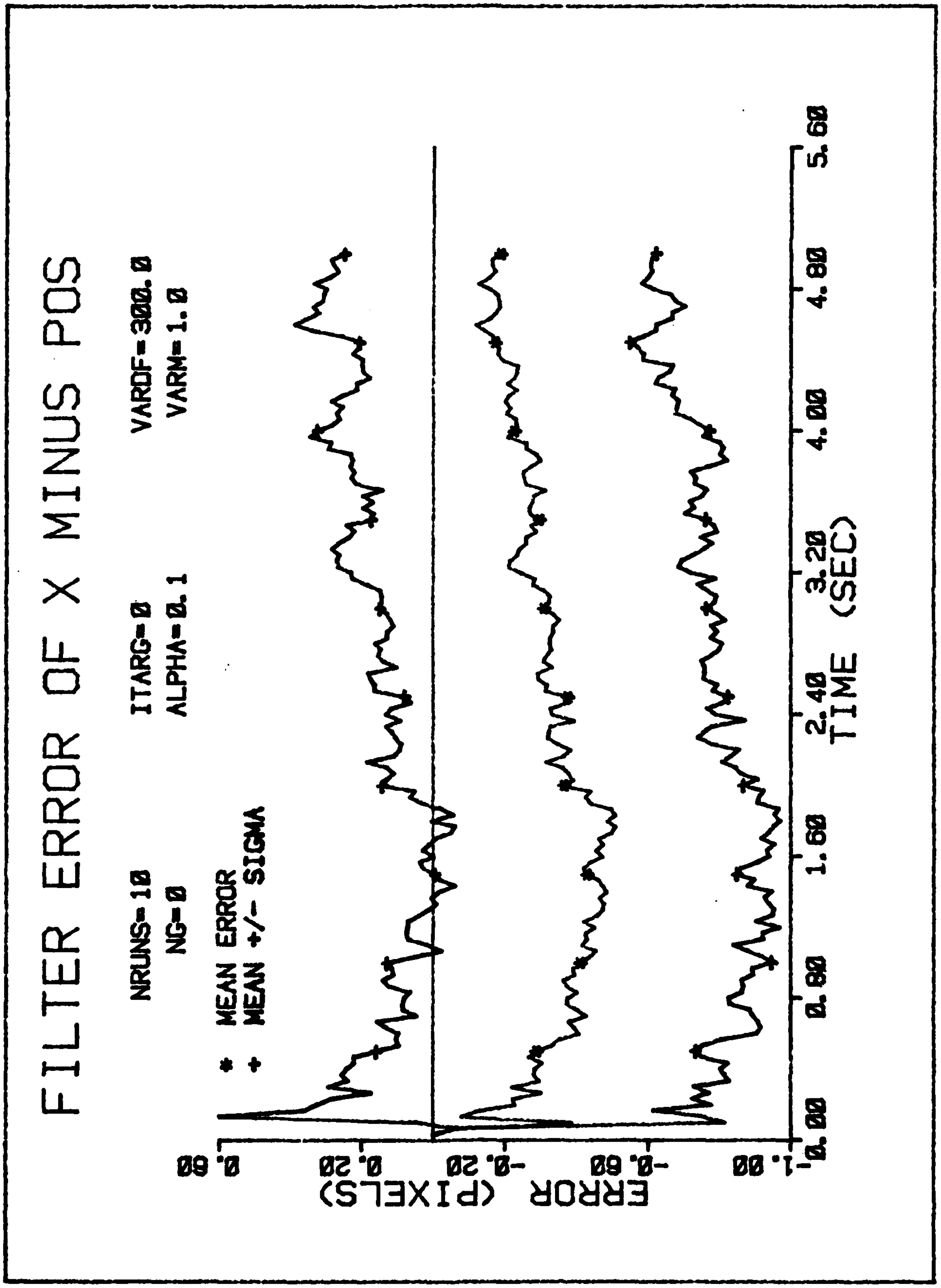


Figure D-37 Case 5 GM Performance Plot

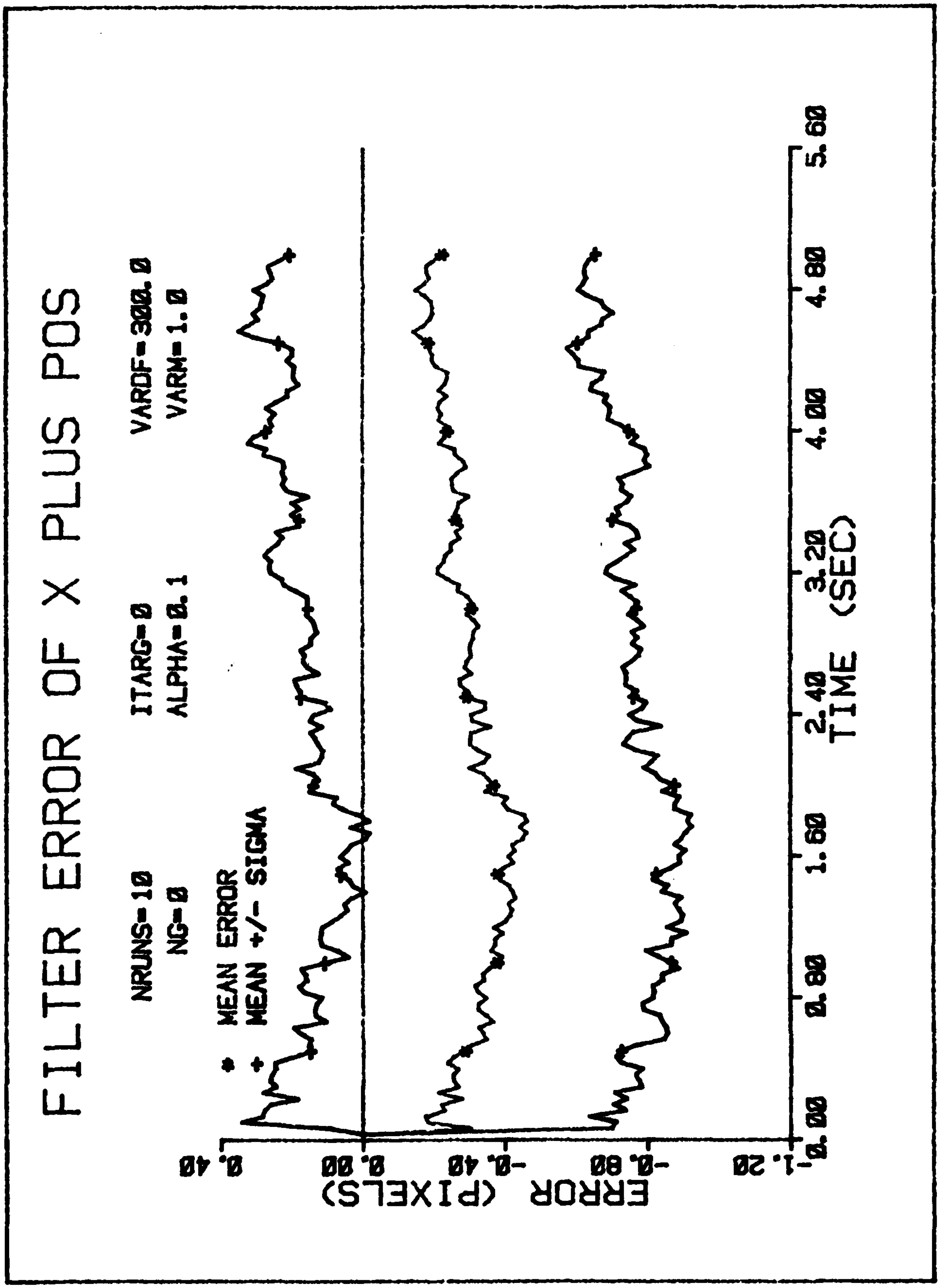


Figure D-38 Case 5 GM Performance Plot

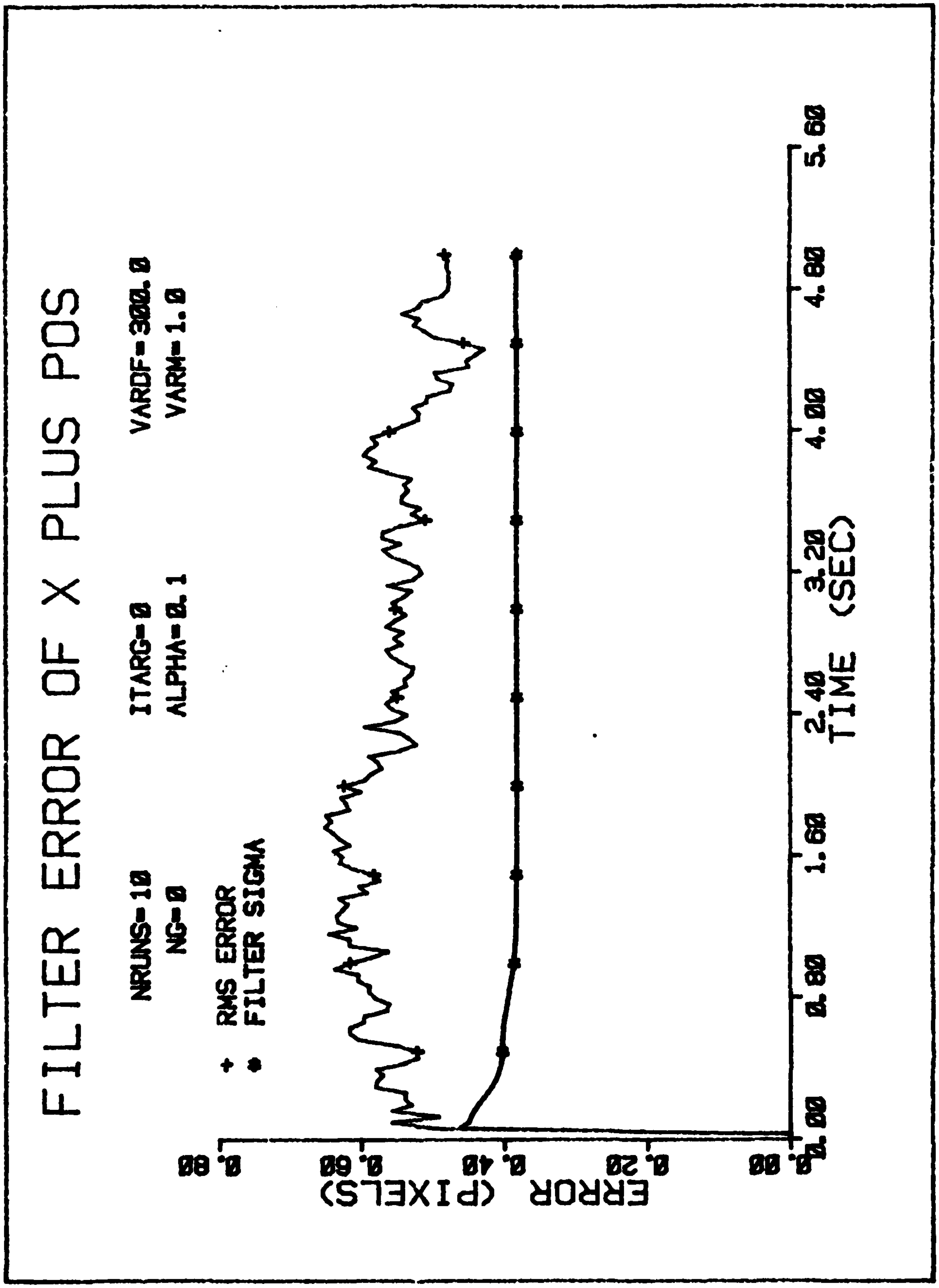


Figure D-39 Case 5 GM Performance Plot

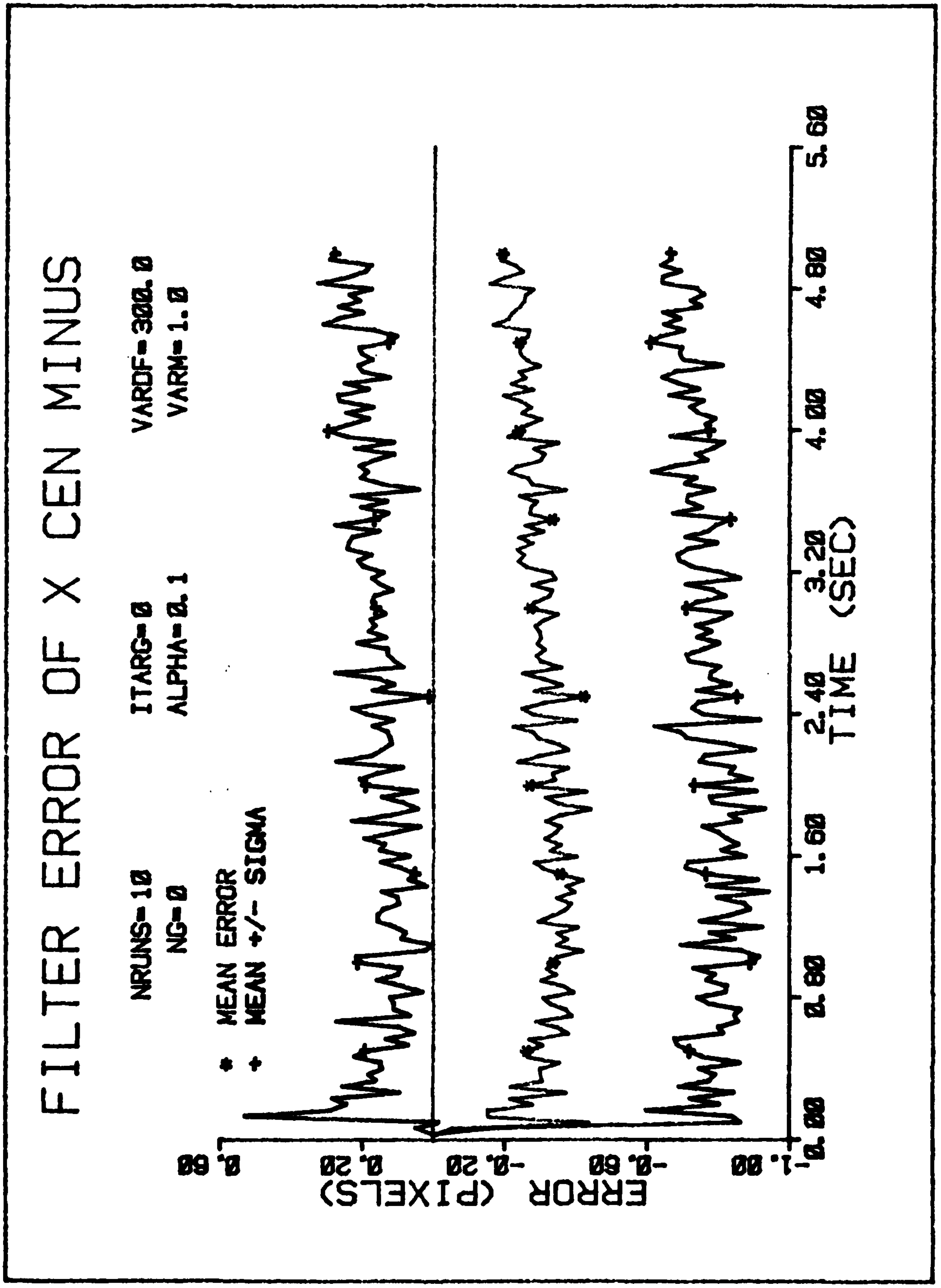


Figure D-40 Case 5 GM Performance Plot

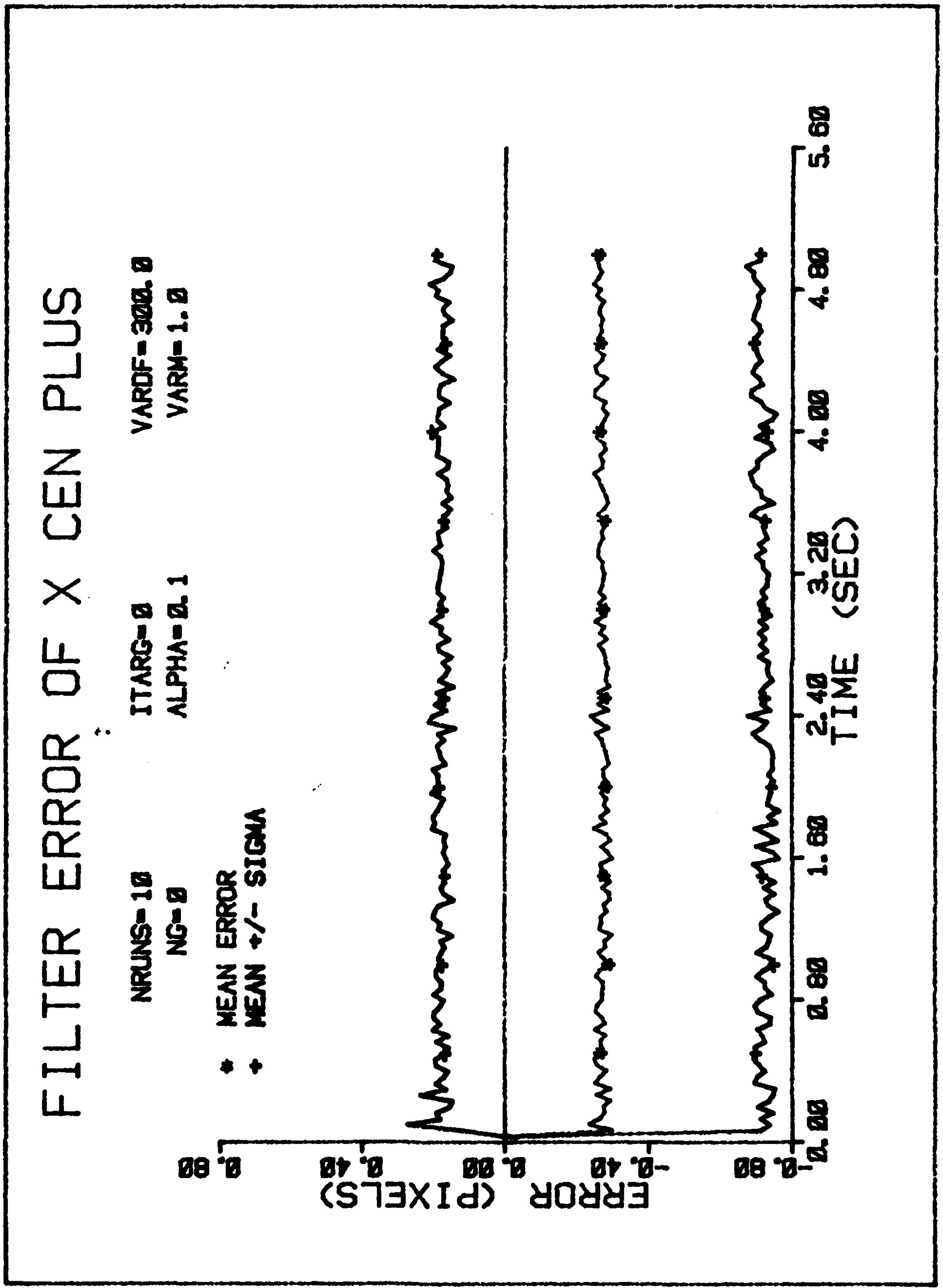


Figure D-41 Case 5 GM Performance Plot

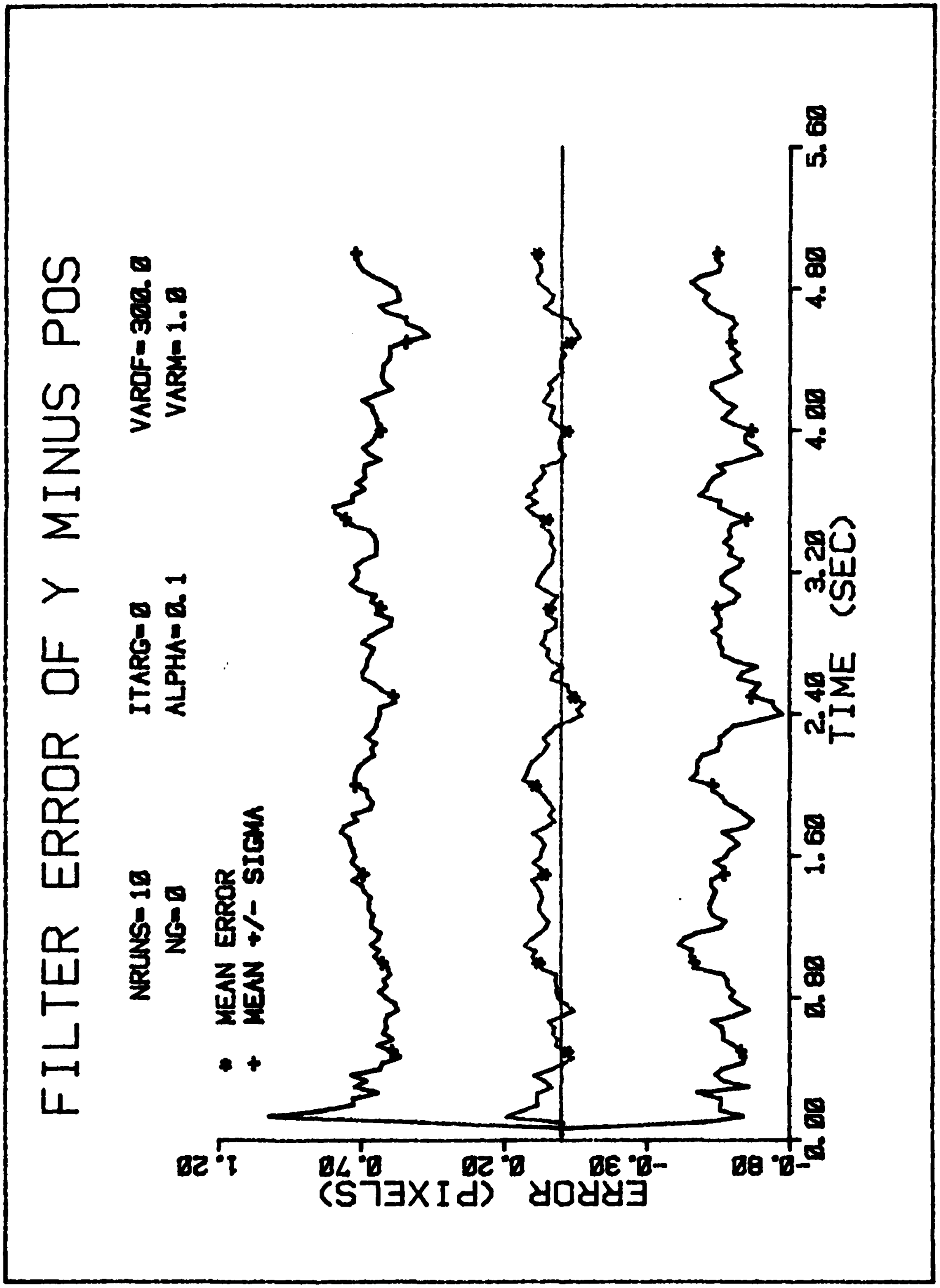


Figure D-42 Case 5 GM Performance Plot

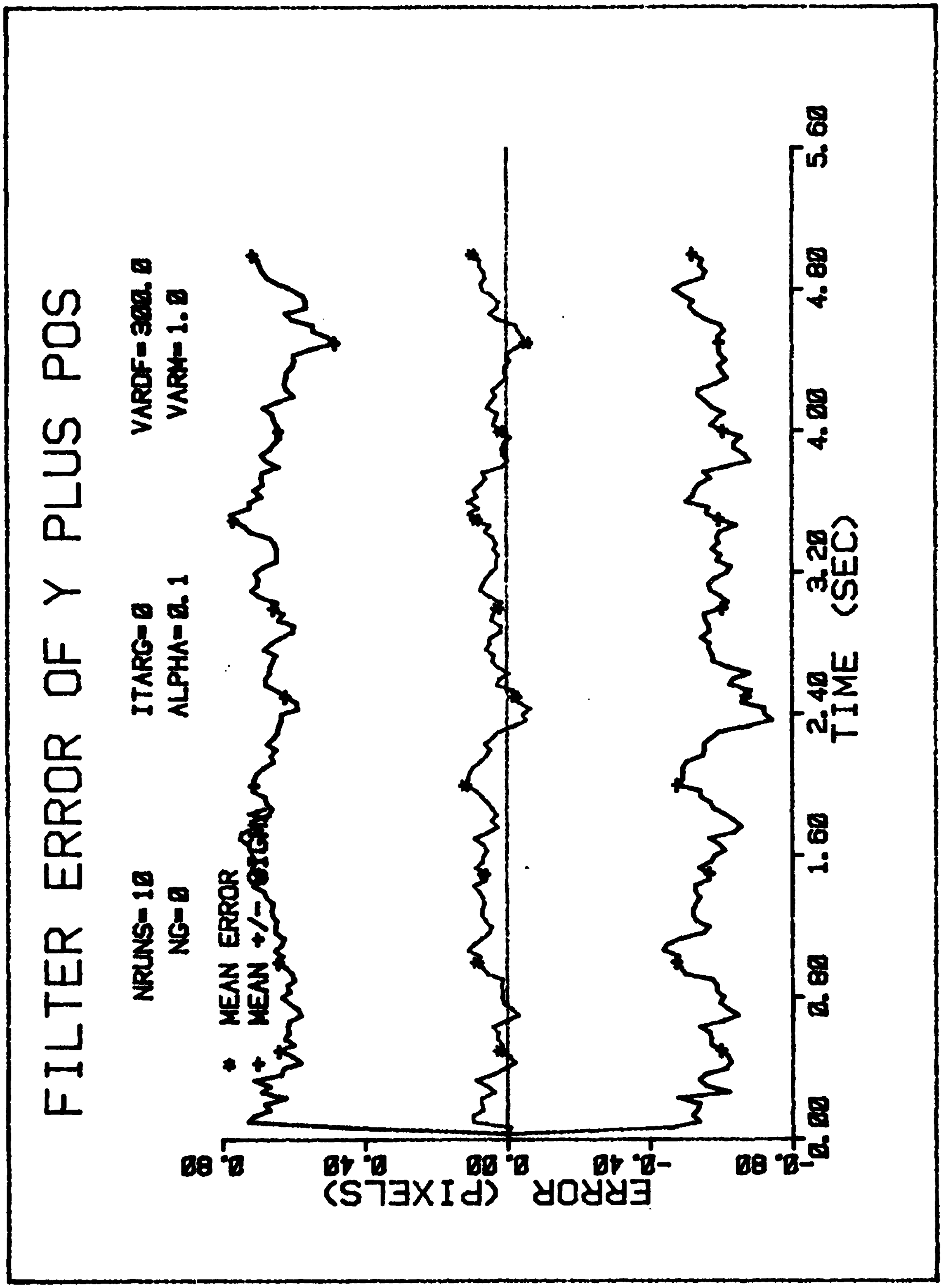


Figure D-43 Case 5 GM Performance Plot

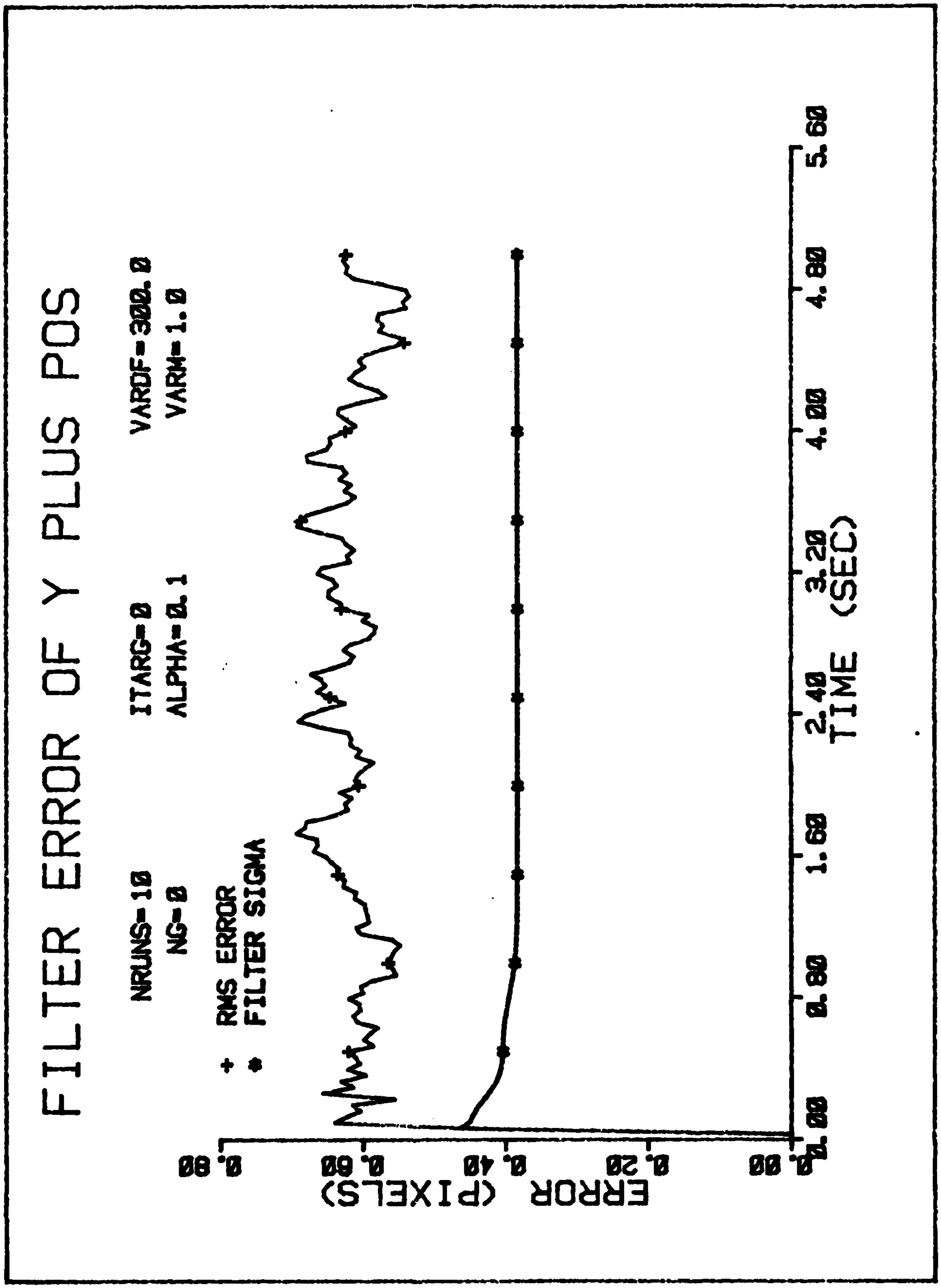


Figure D-44 Case 5 GM Performance Plot

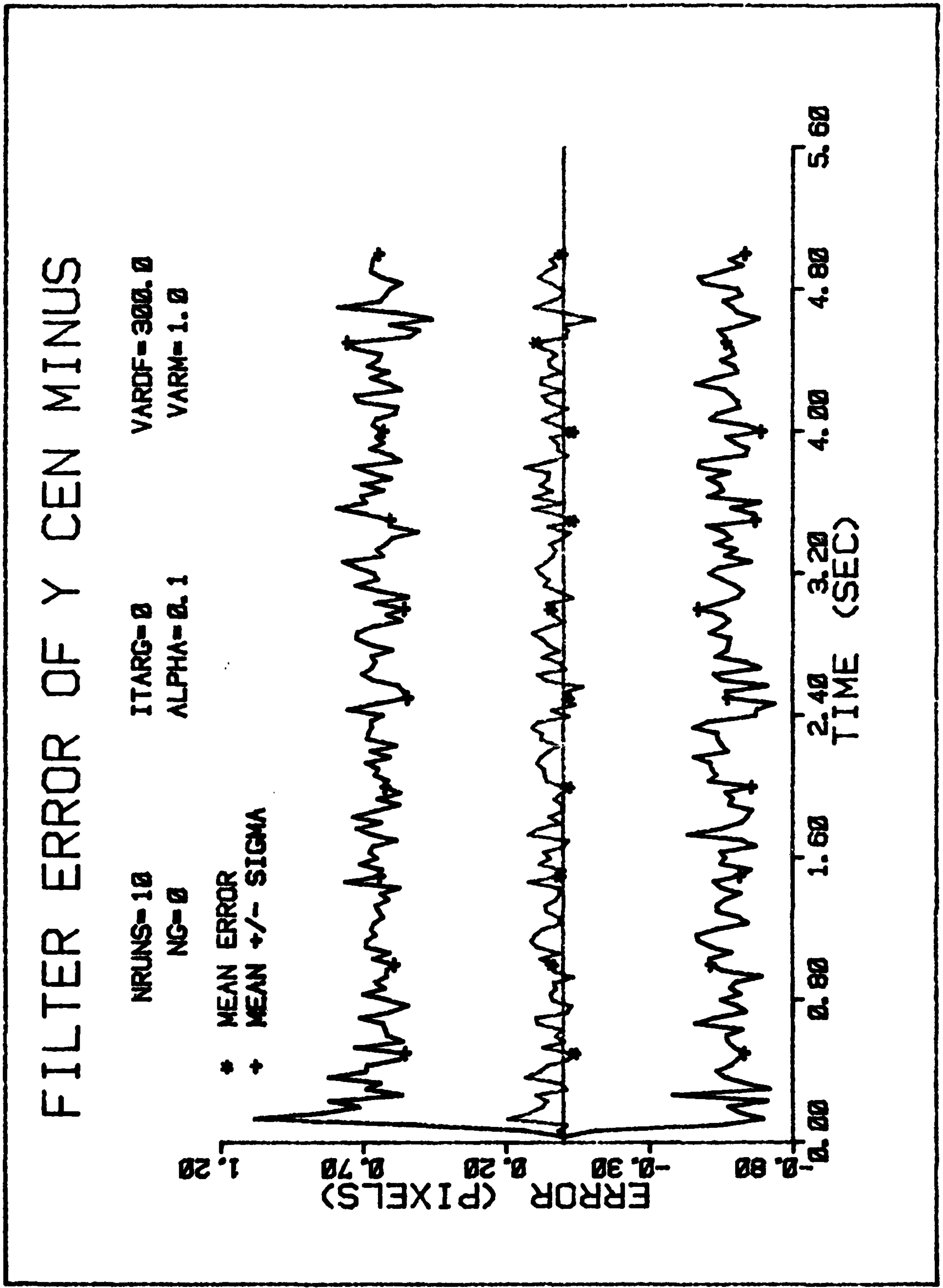


Figure D-45 Case 5 GM Performance Plot

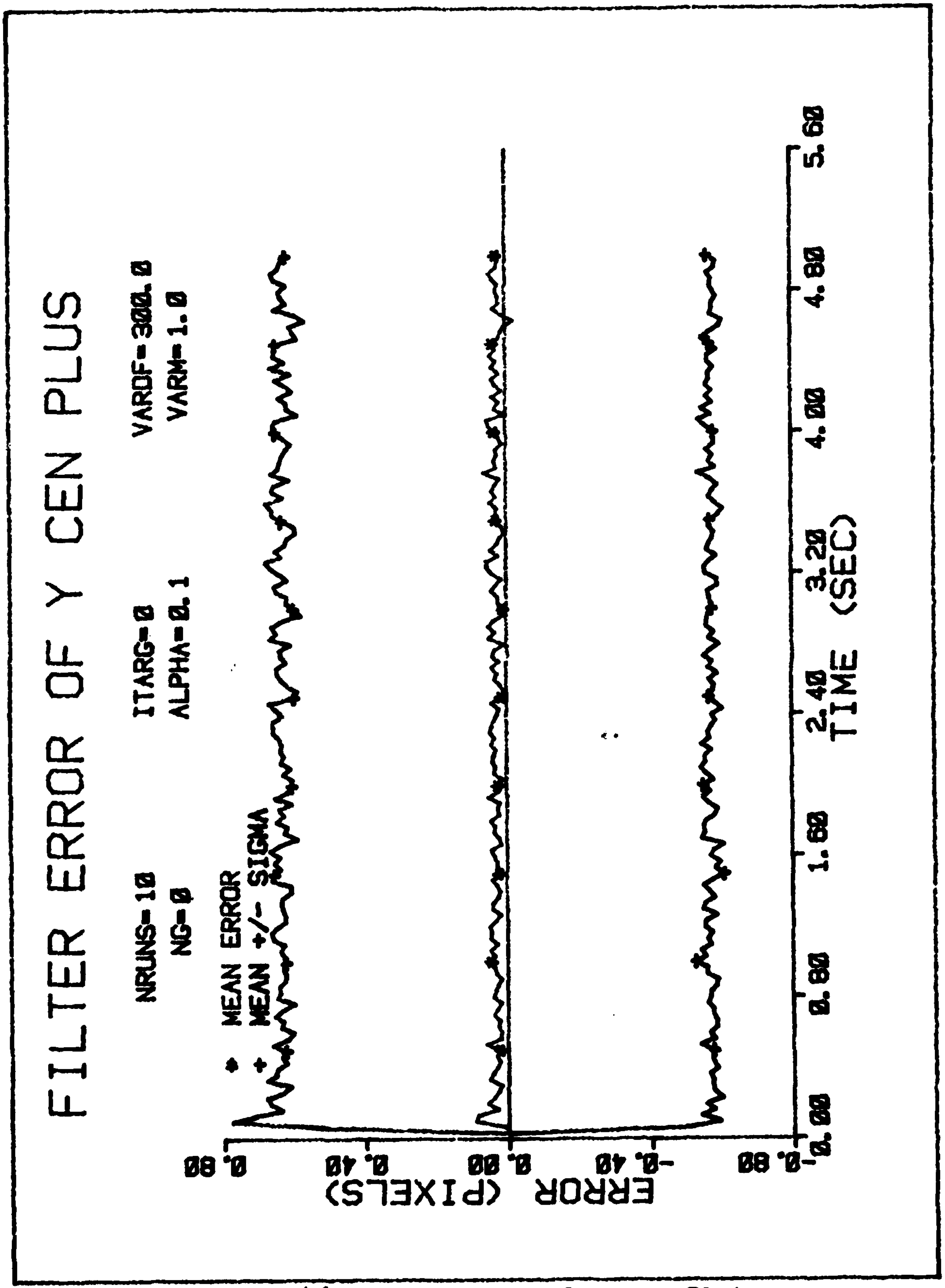


Figure D-46 Case 5 GM Performance Plot

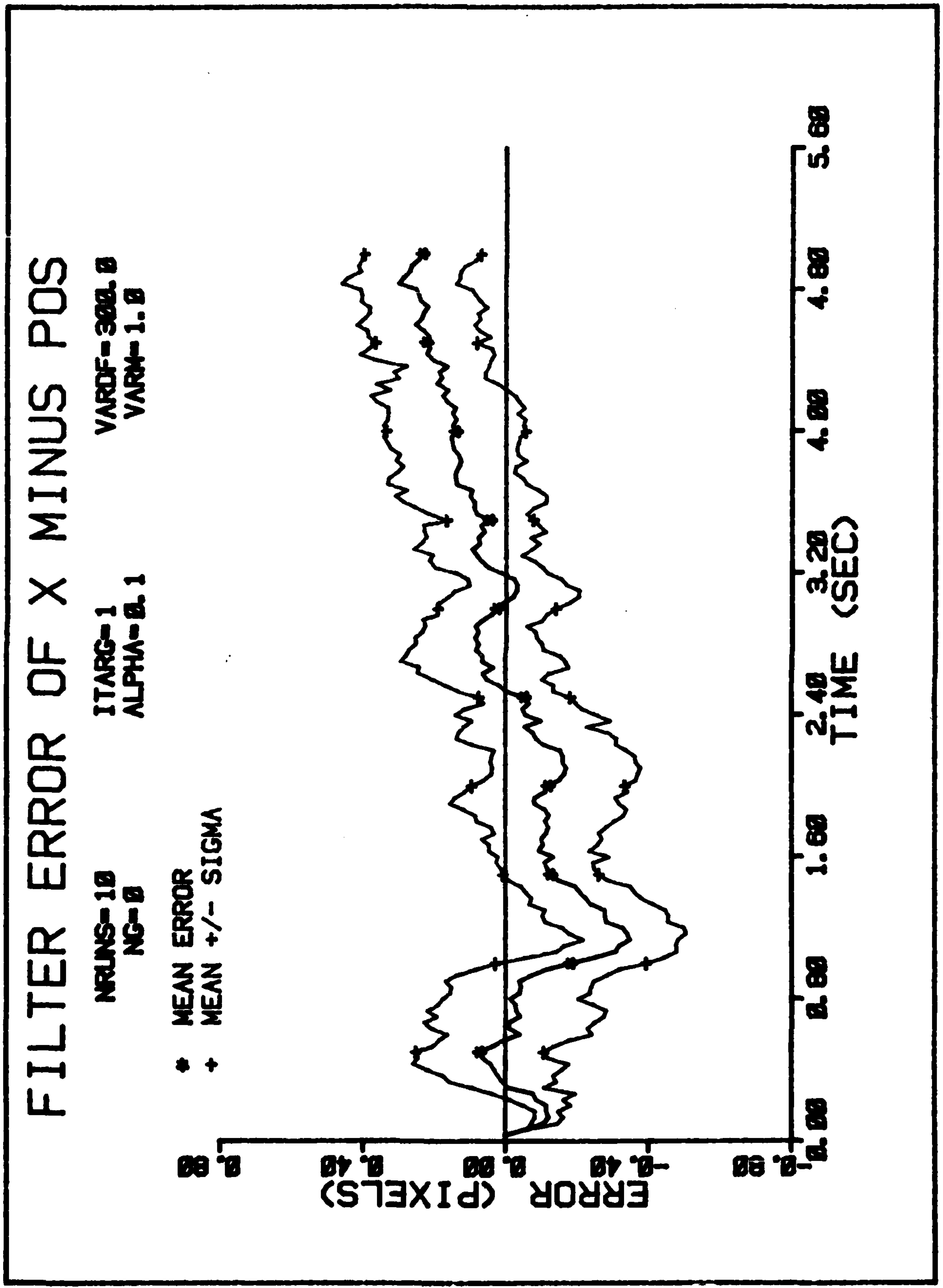


Figure D-47 Case 6 GM Performance Plot

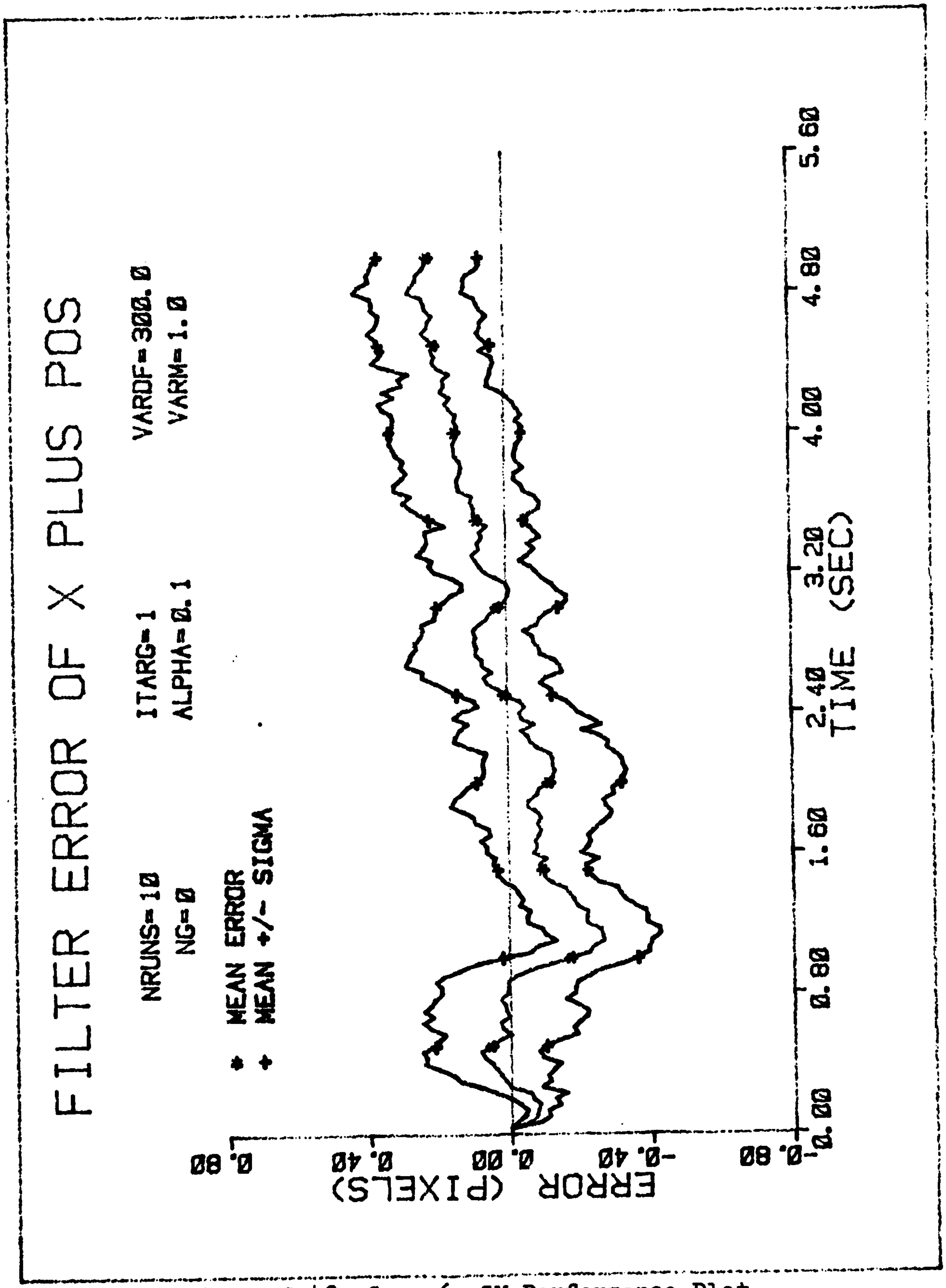


Figure D-48 Case 6 GM Performance Plot

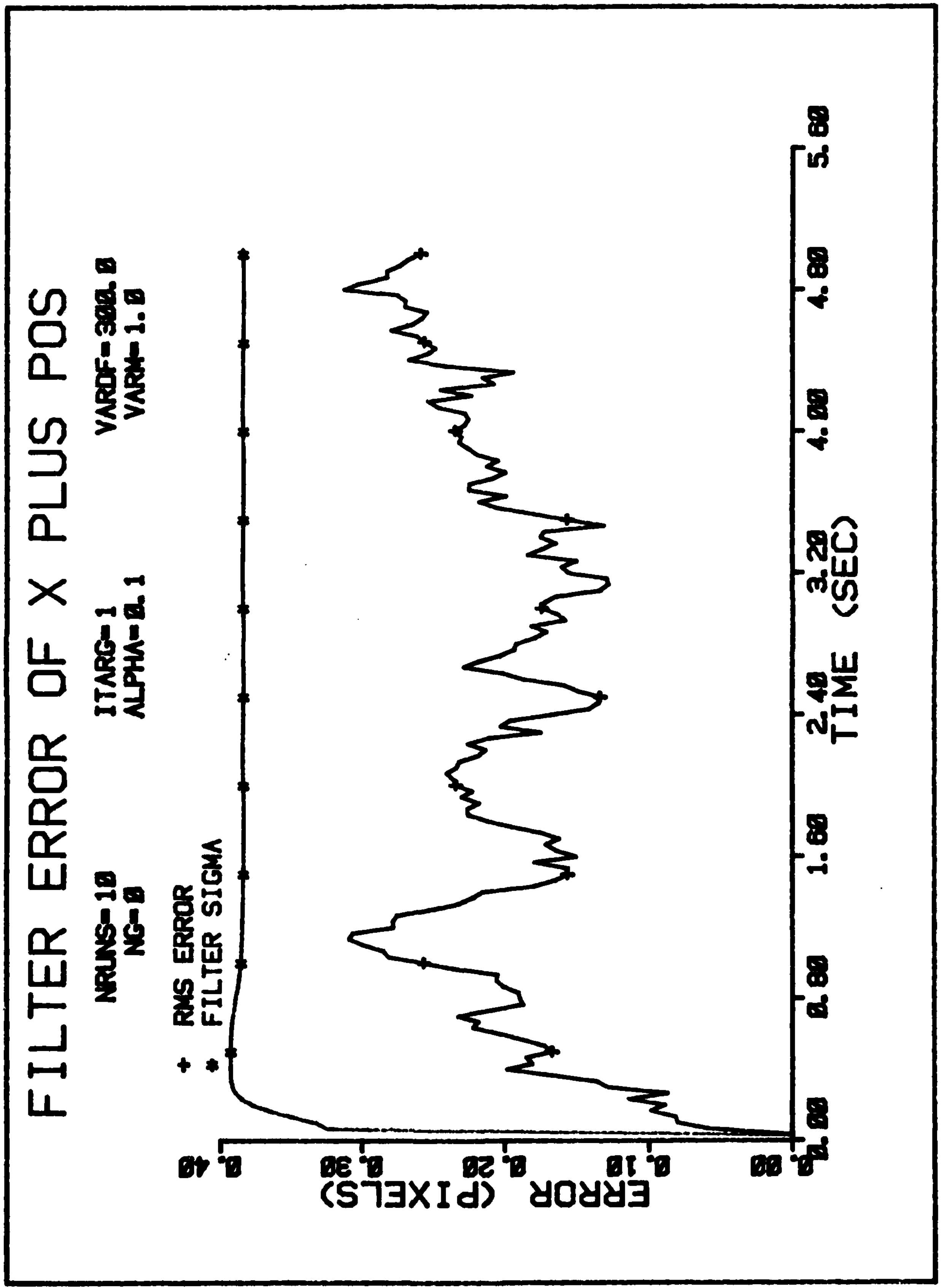


Figure D-49 Case 6 GM Performance Plot

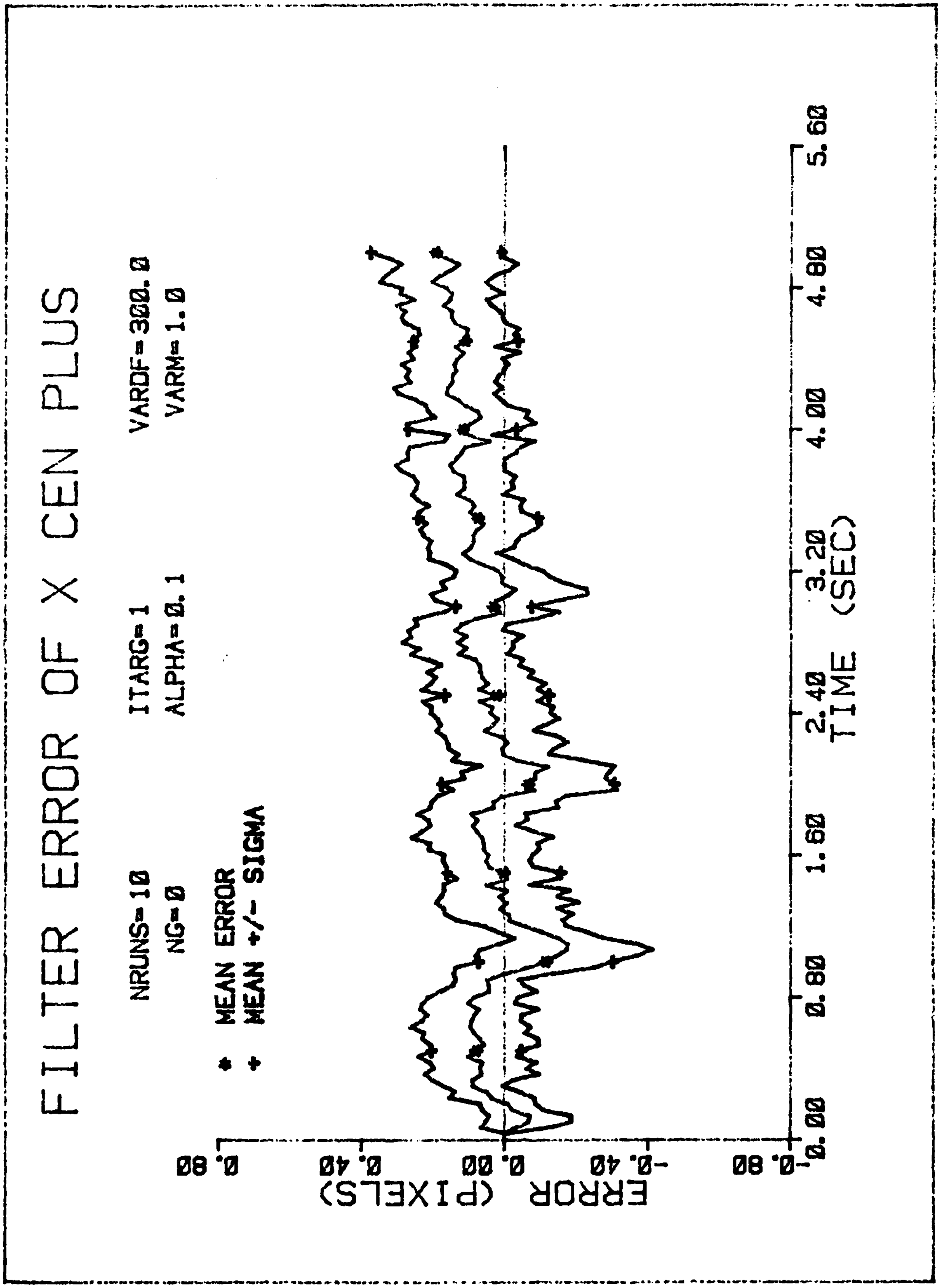


Figure D-50 Case 6 GM Performance Plot

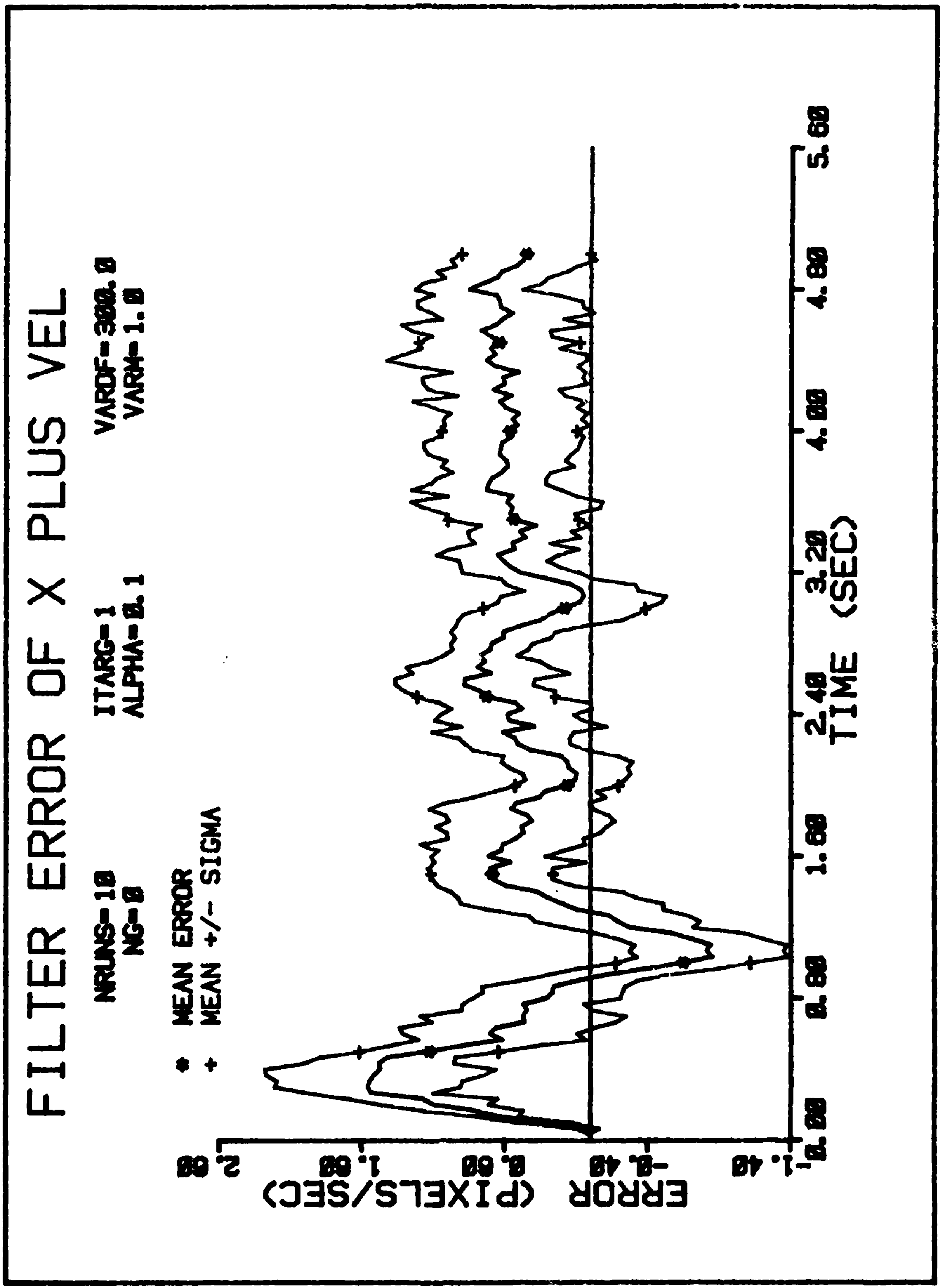


Figure D-51 Case 6 GM Performance Plot

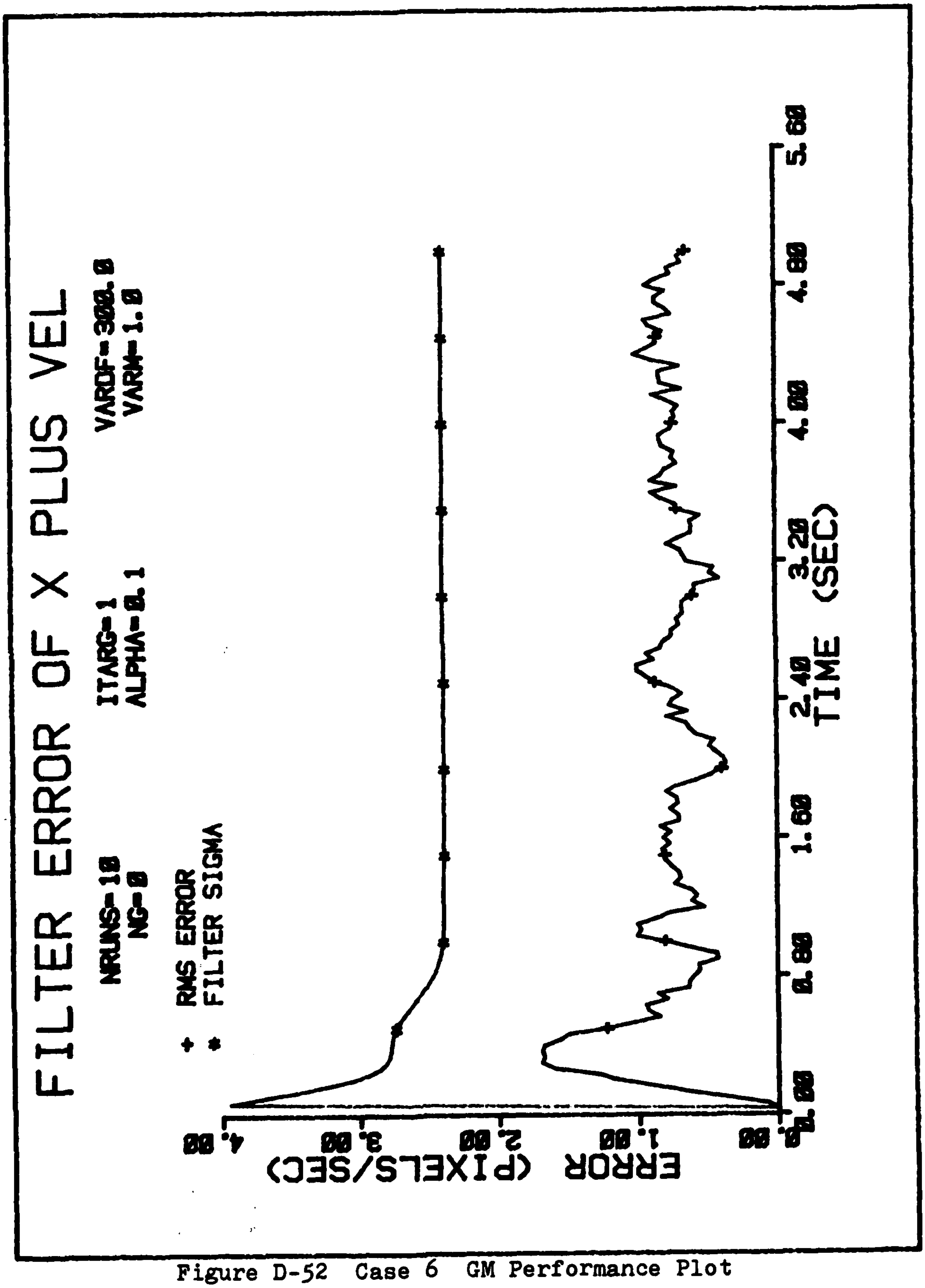


Figure D-52 Case 6 GM Performance Plot

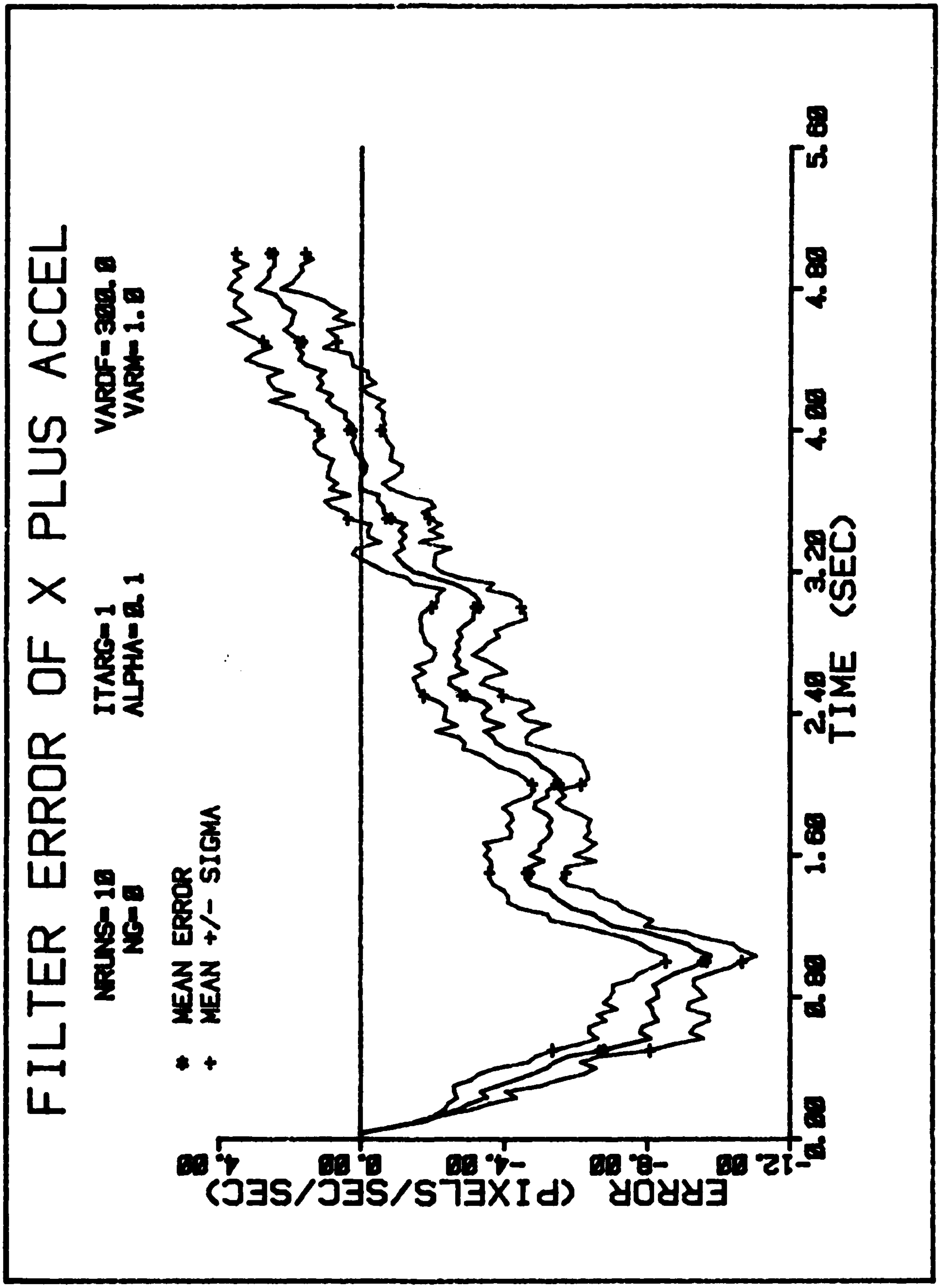


Figure D-53 Case 6 GM Performance Plot

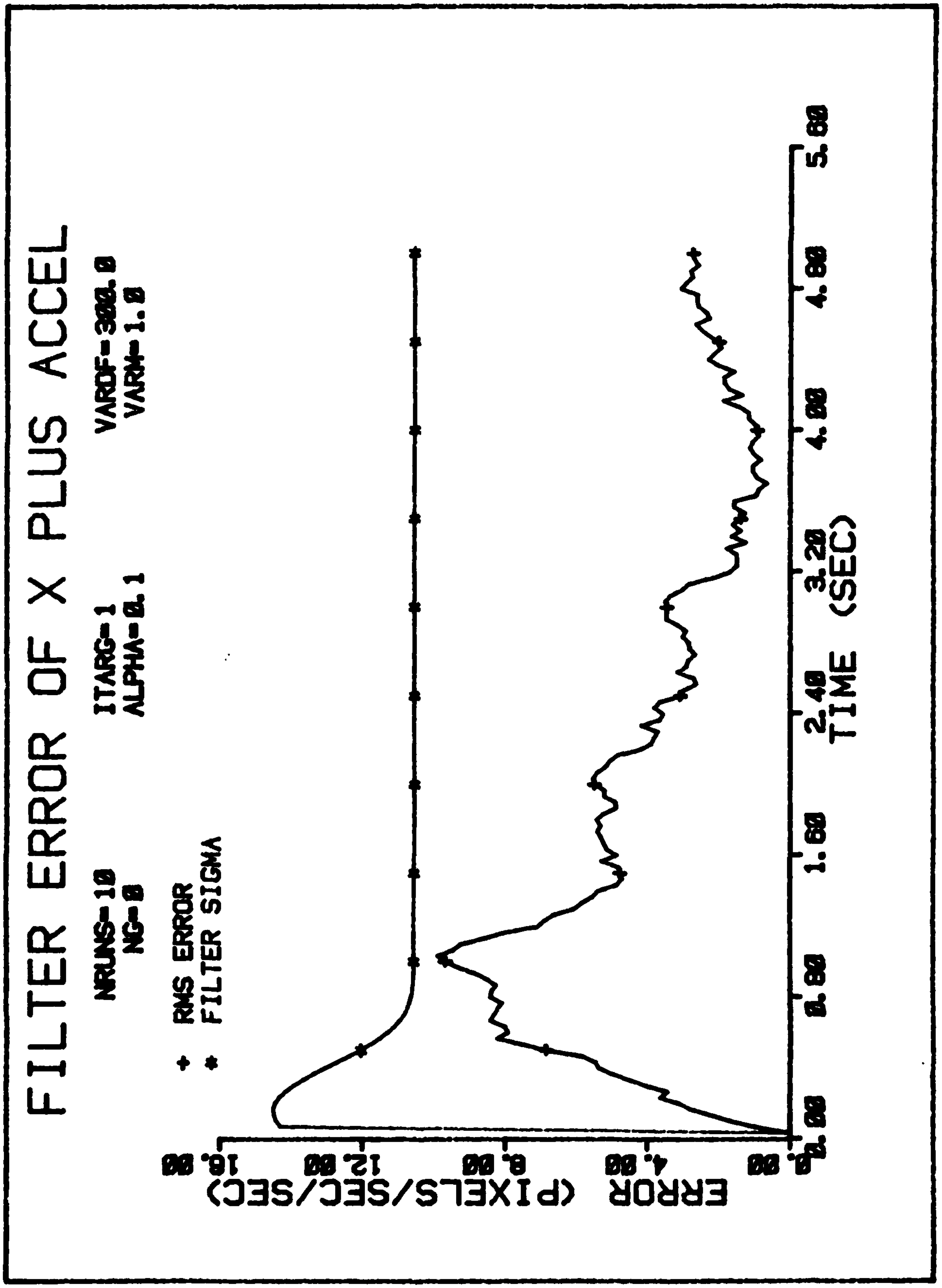


Figure D-54 Case 6 GM Performance Plot

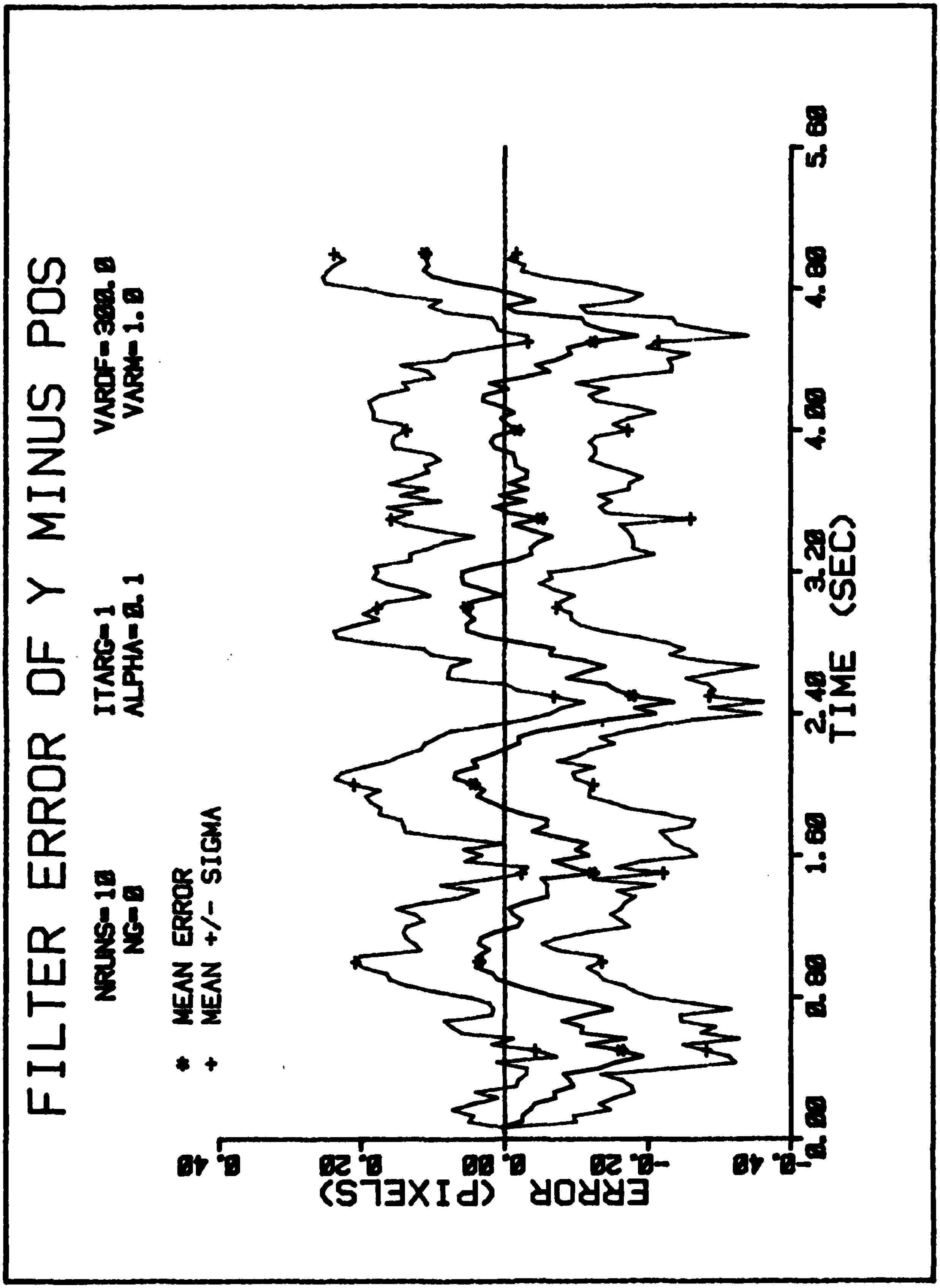


Figure D-55 Case 6 Gm Performance Plot

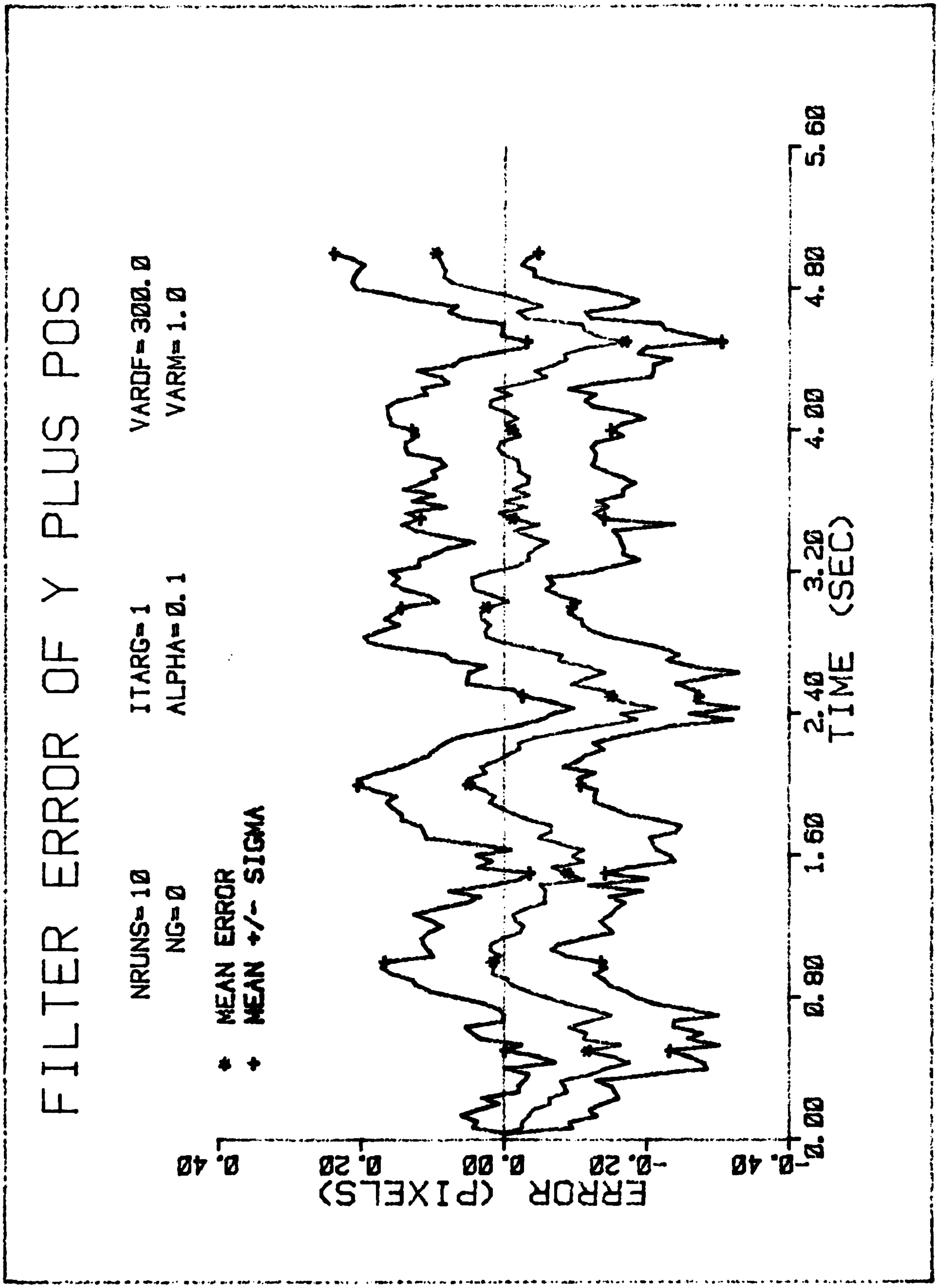


Figure D-56 Case 6 GM Performance Plot

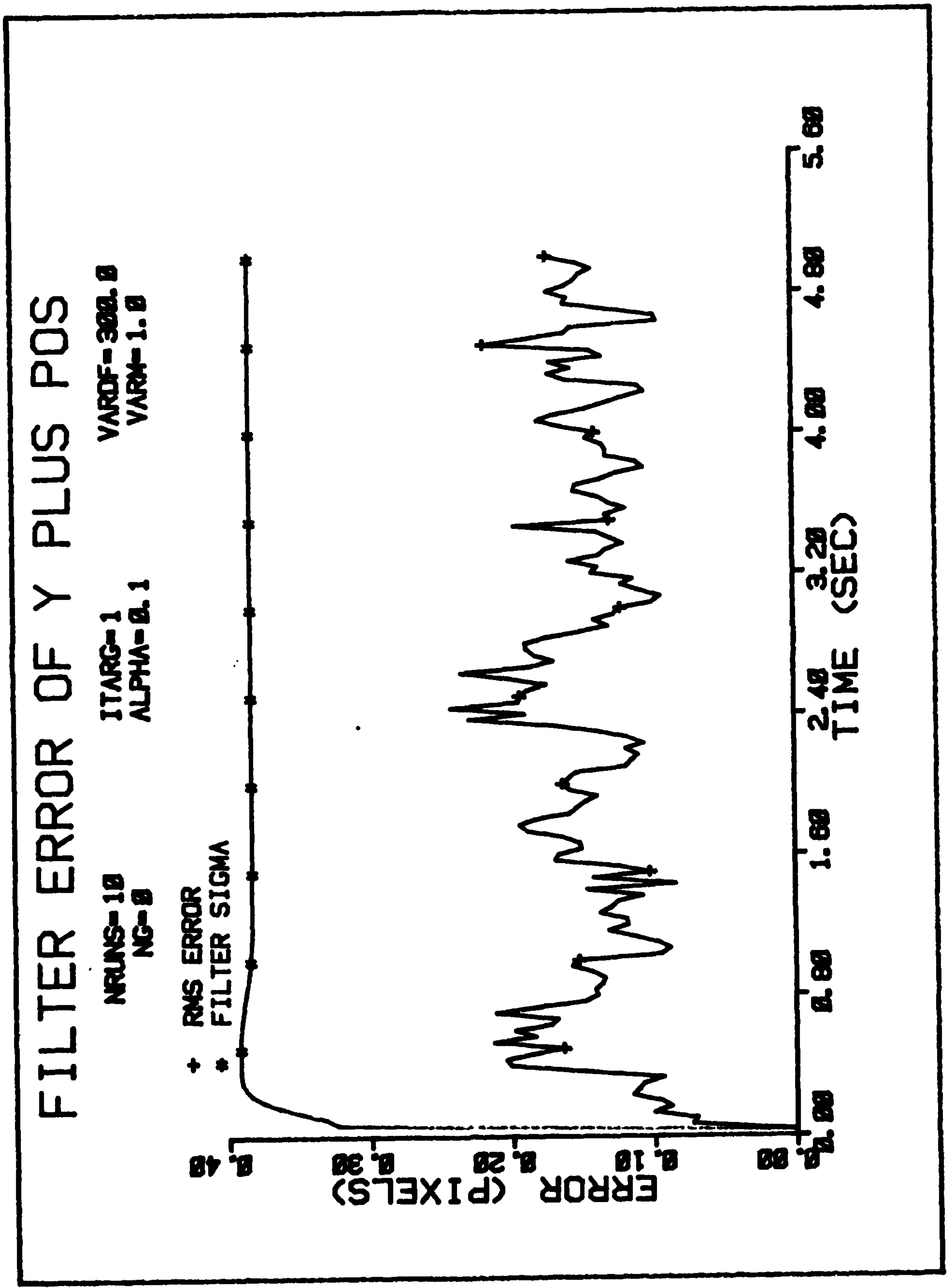


Figure D-57 Case 6 GM Performance Plot

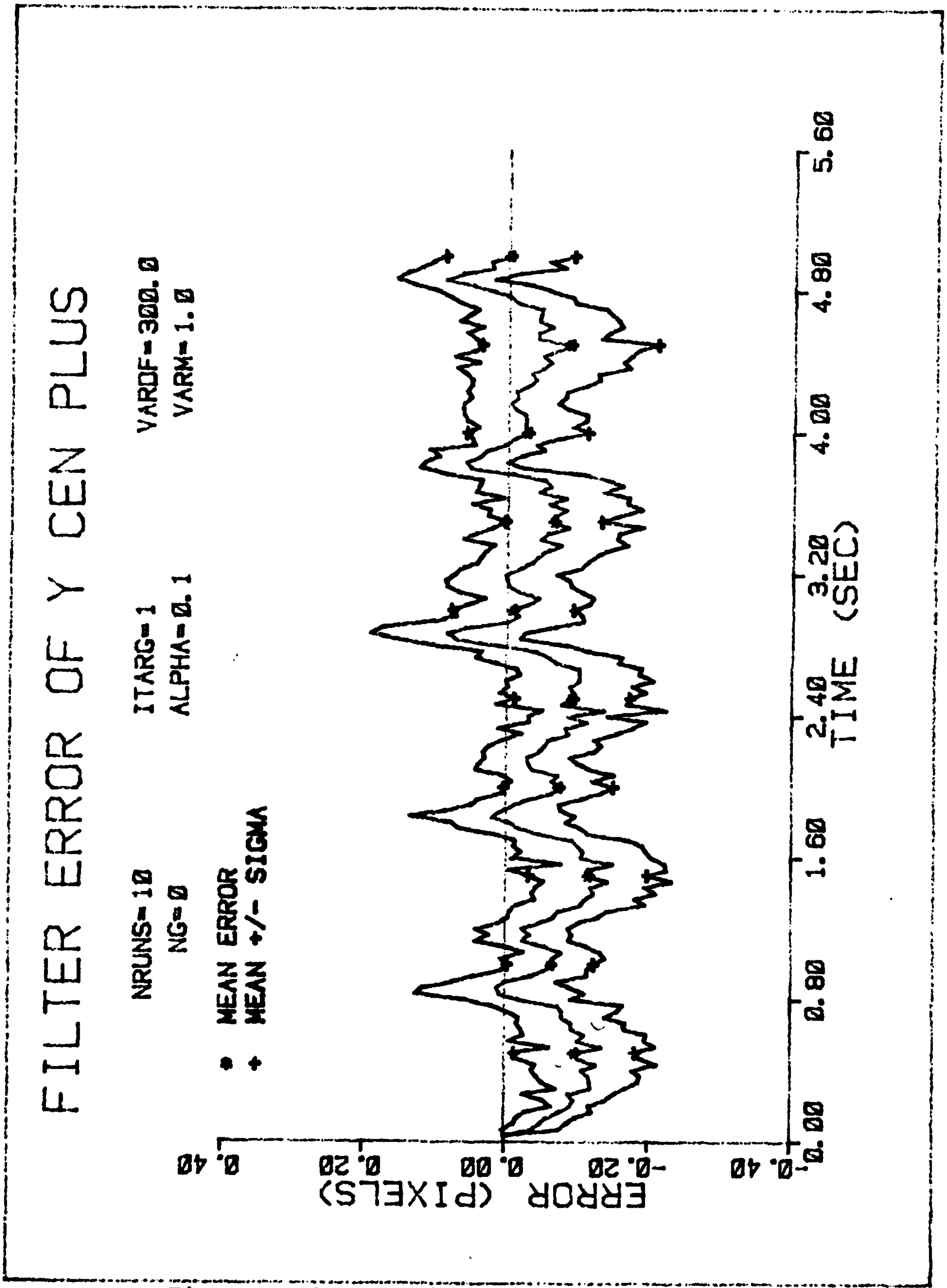


Figure D-58 Case 6 GM Performance Plot

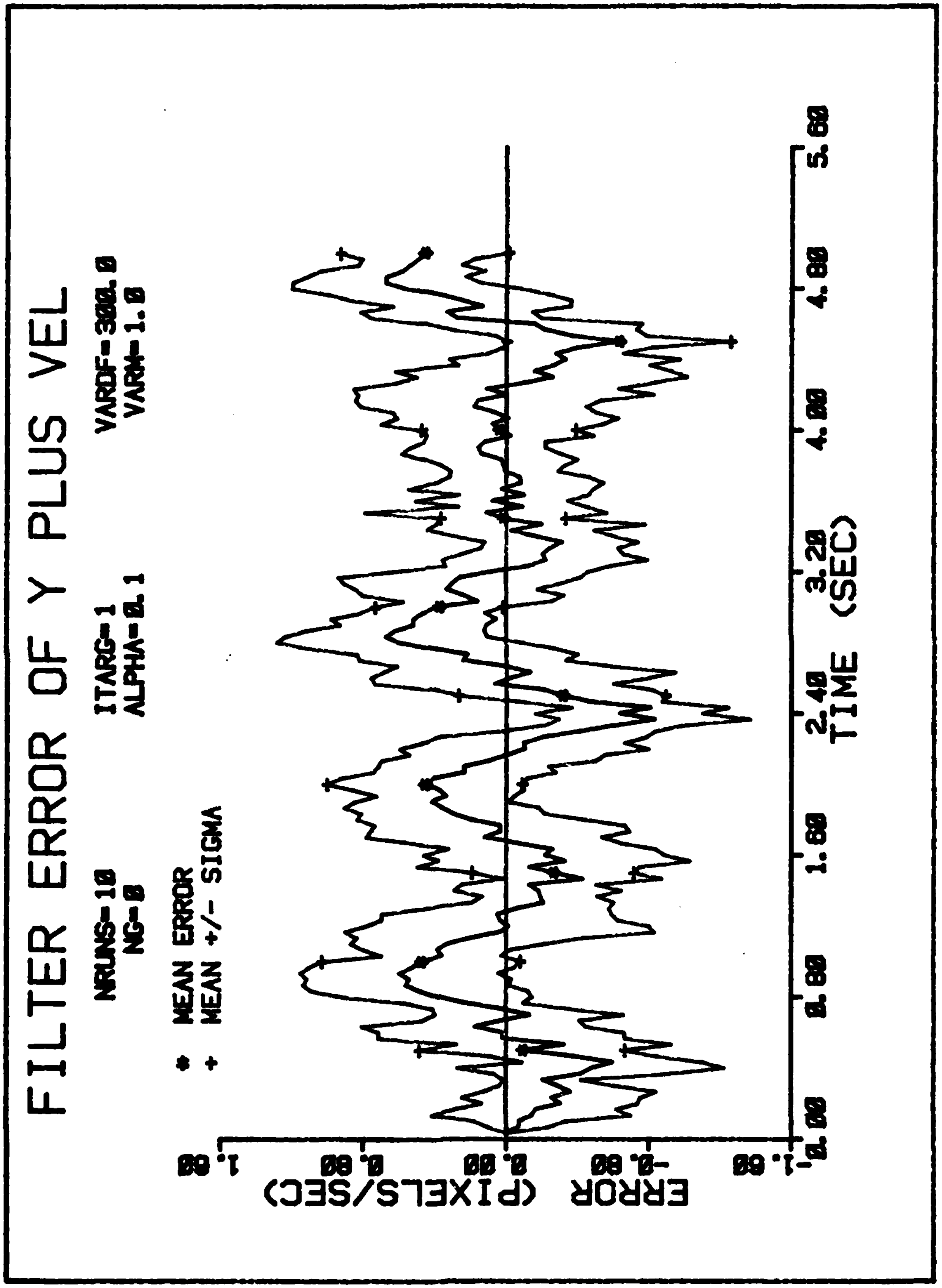


Figure D-59 Case 6 GM Performance Plot

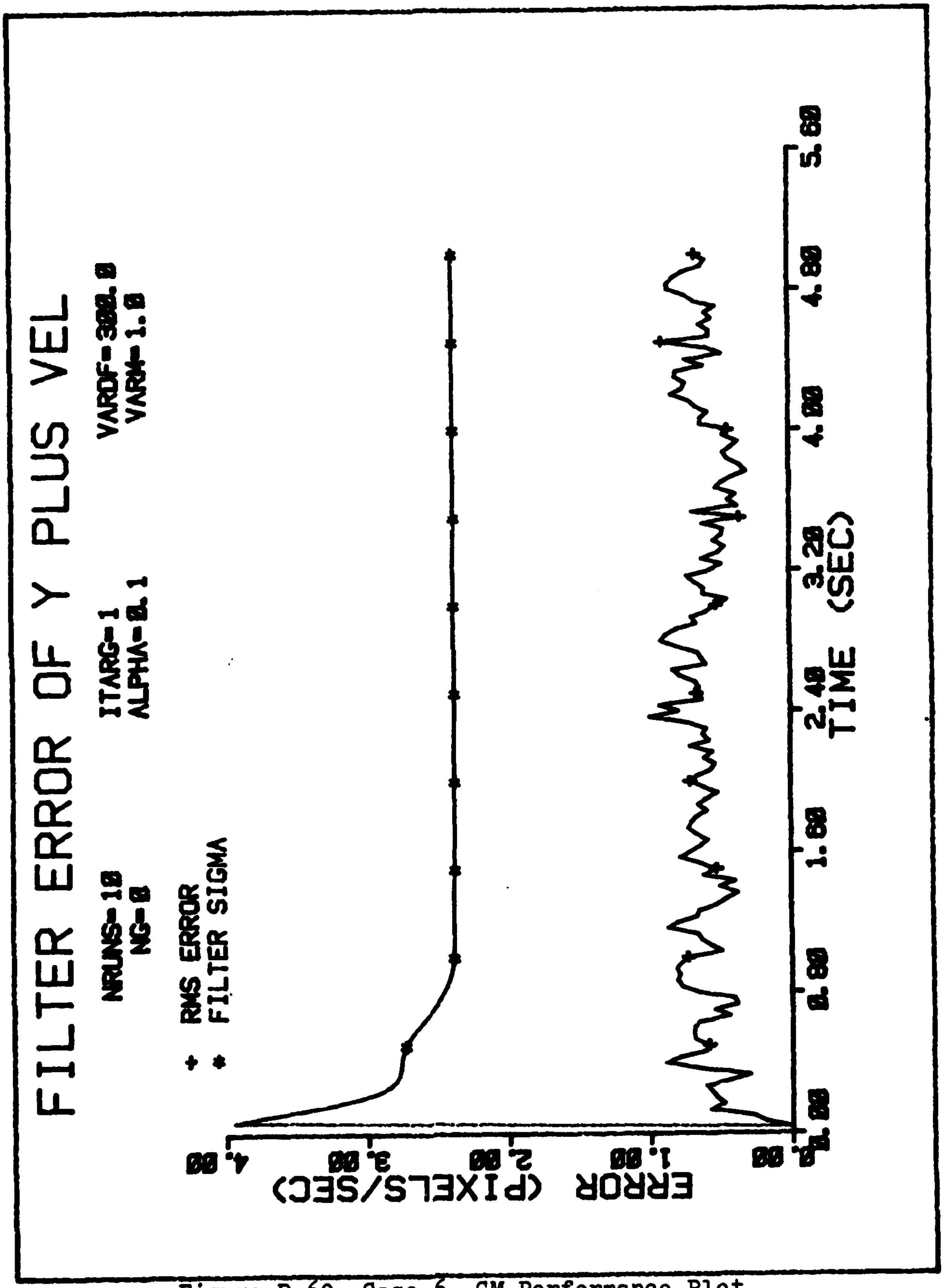


Figure D-60 Case 6 GM Performance Plot

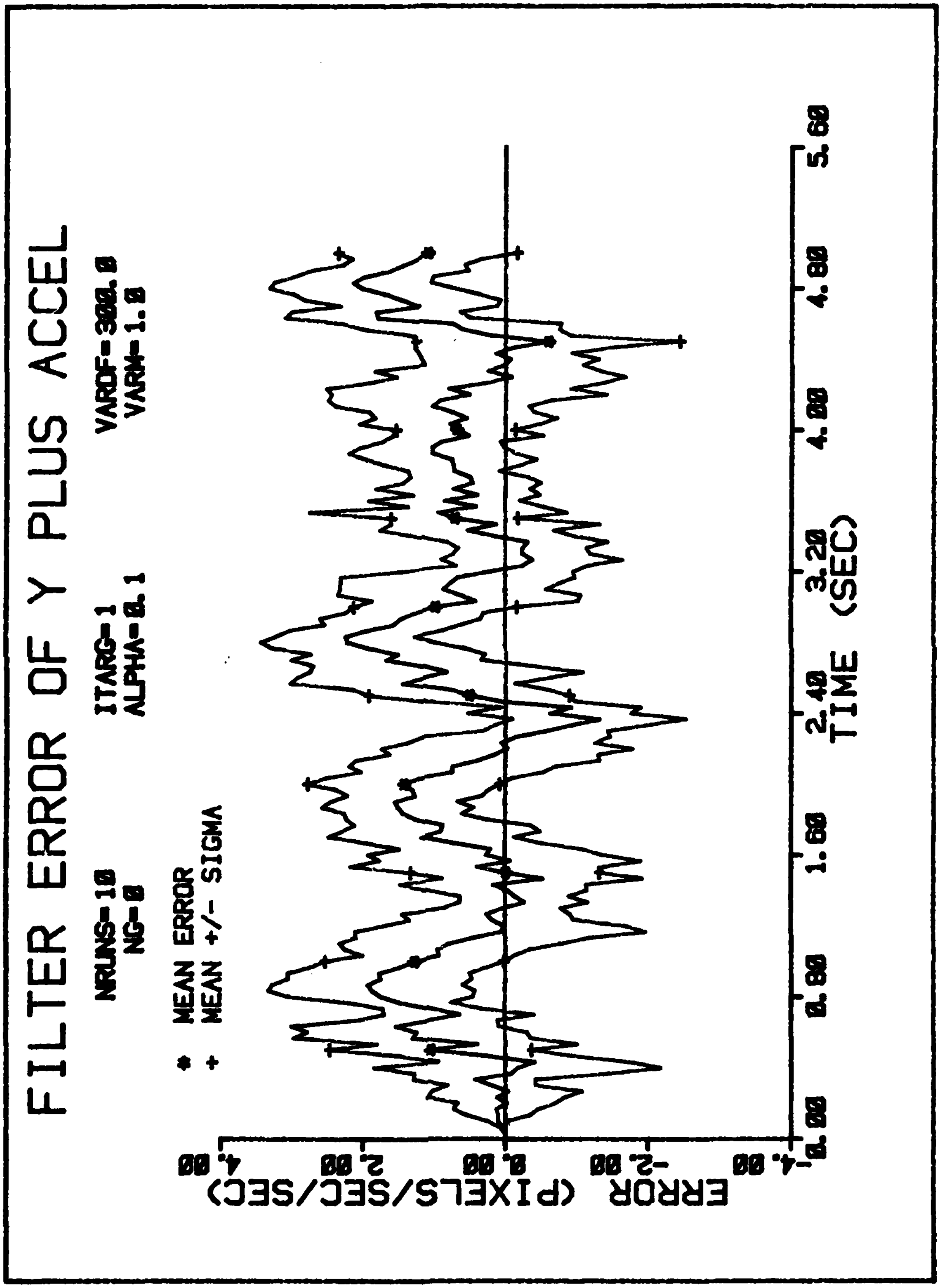


Figure D-61 Case 6 GM Performance Plot

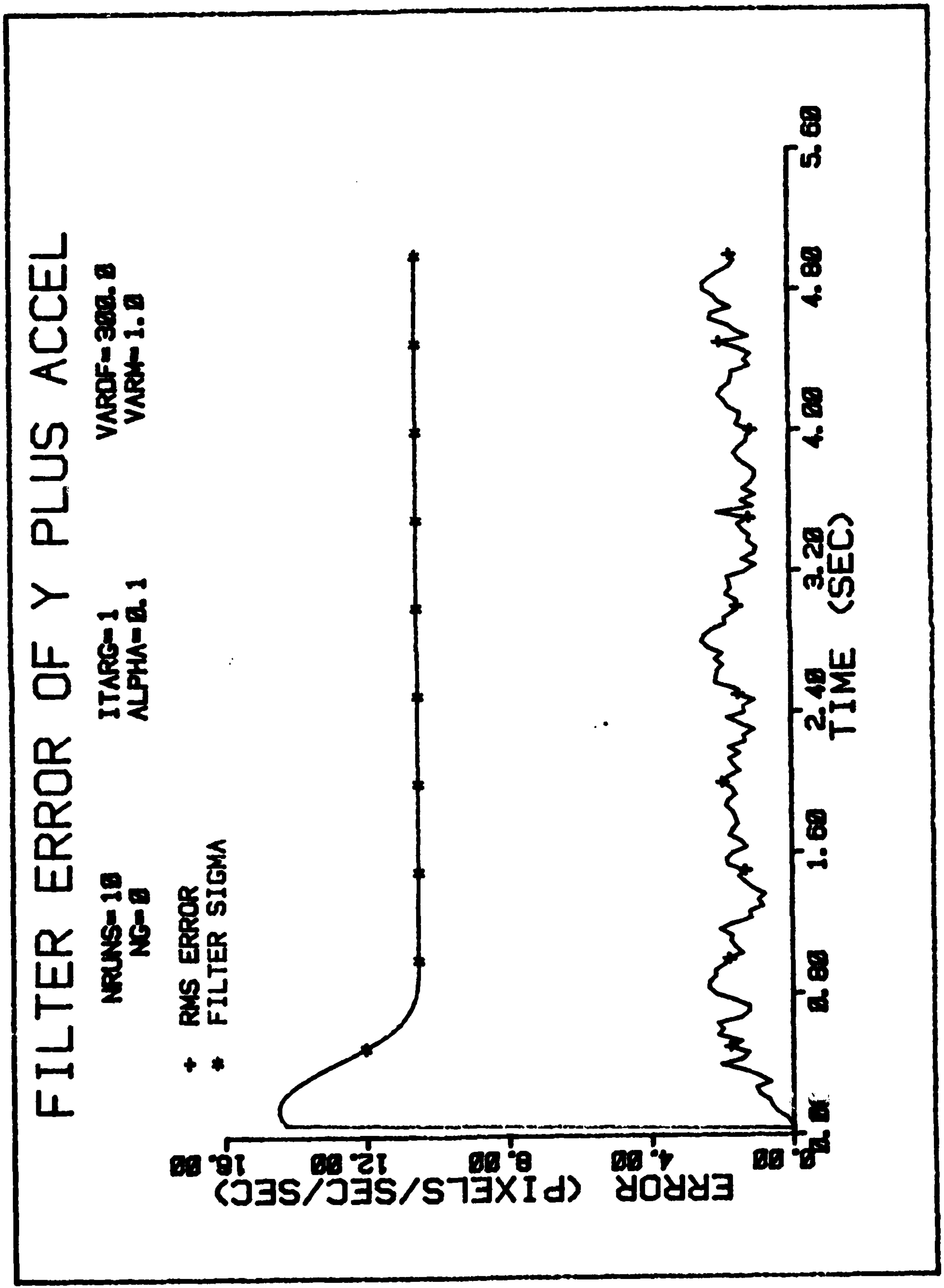


Figure D-62 Case 6 GM Performance Plot

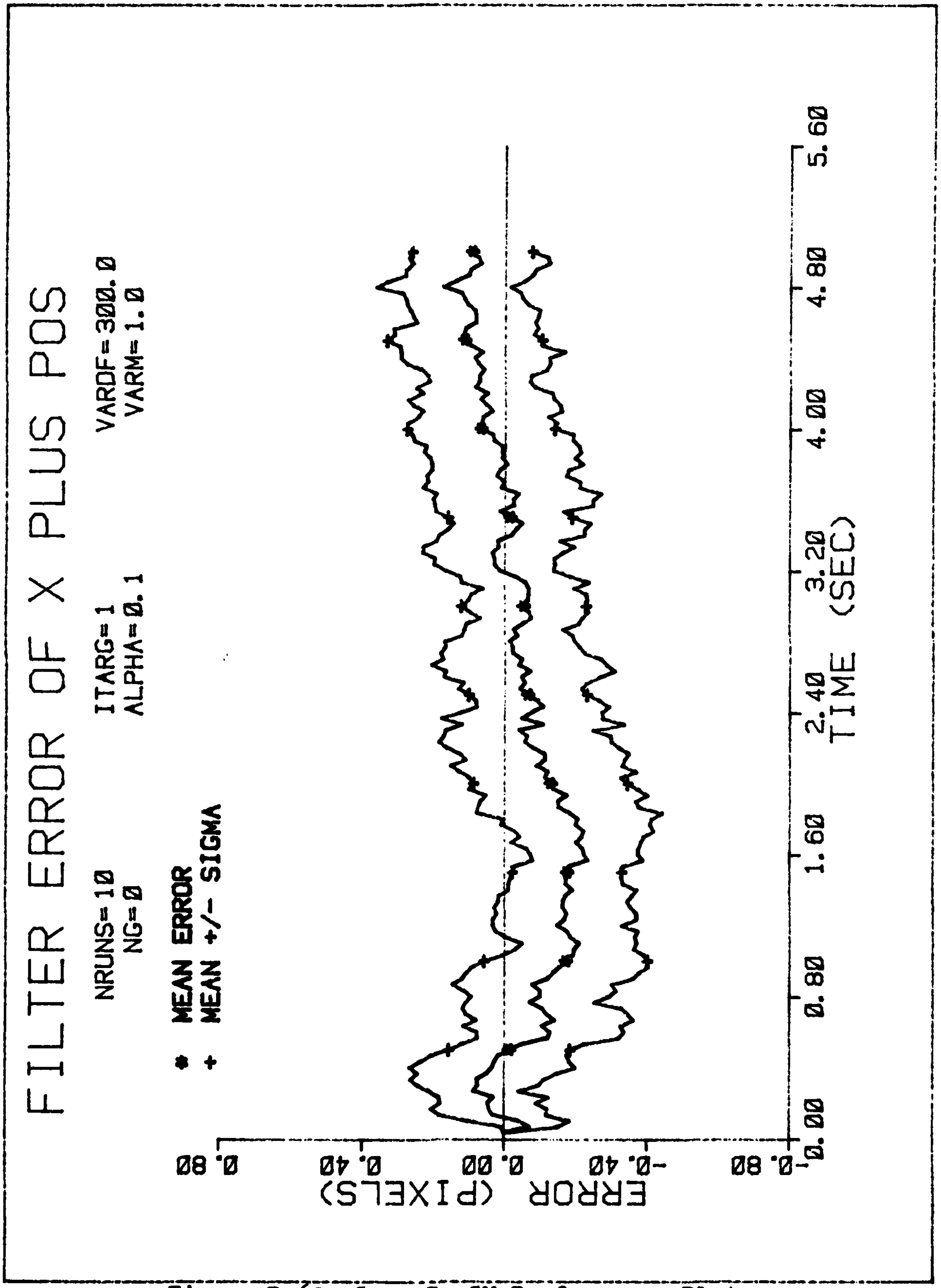


Figure D-63 Case 7 GM Performance Plot

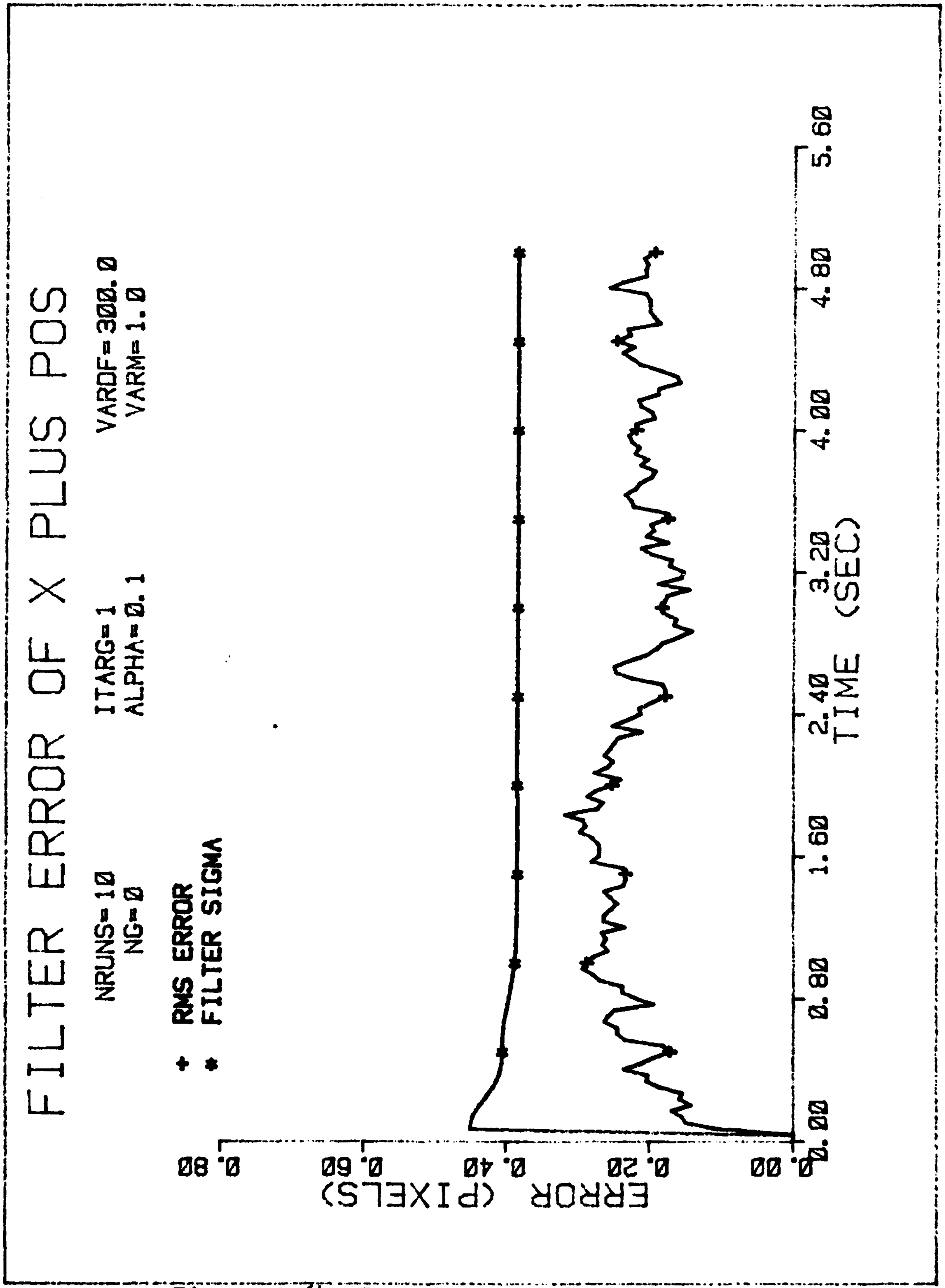


Figure D-64 Case 7 GM Performance Plot

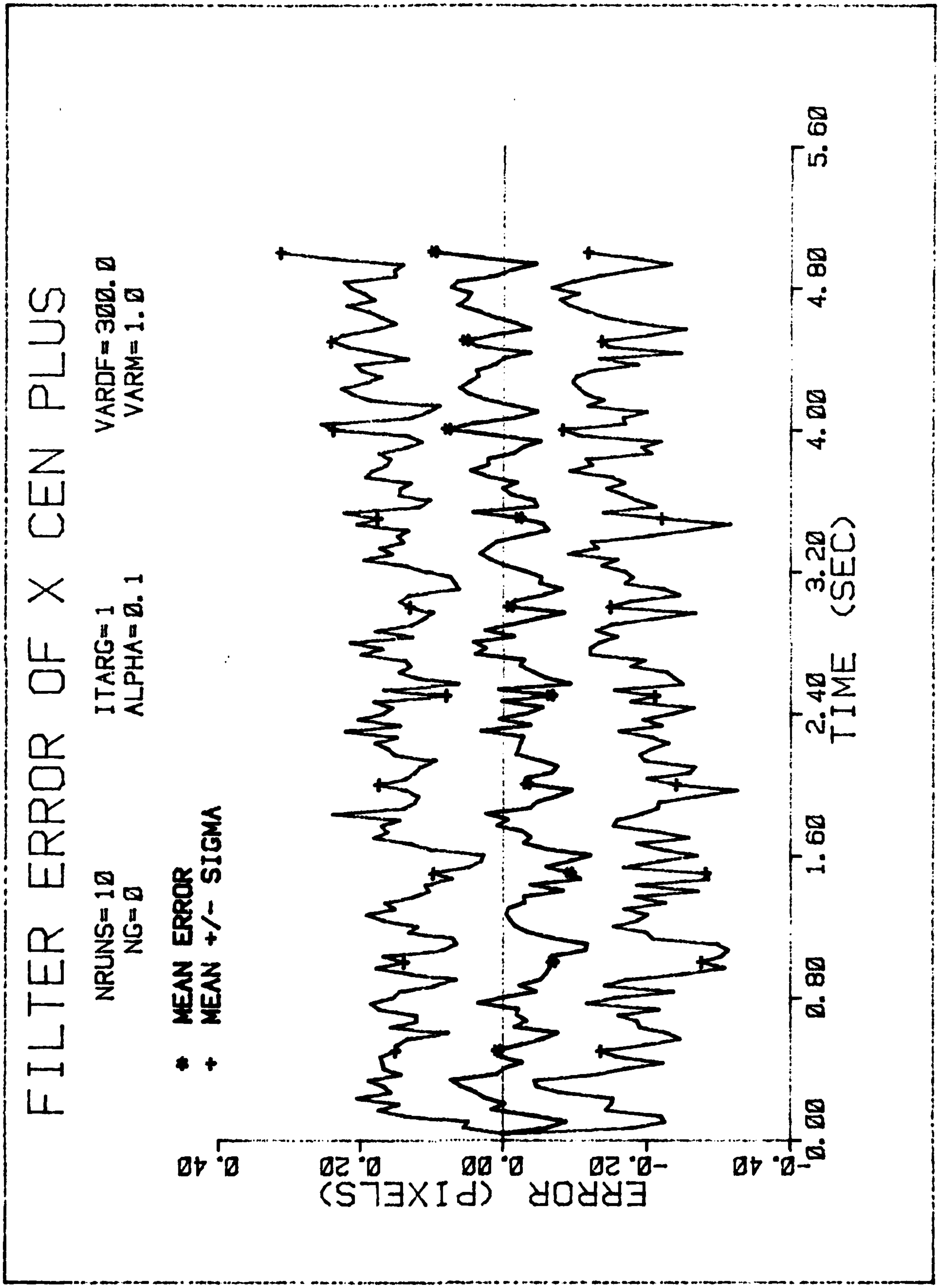


Figure D-65 Case 7 GM Performance Plot

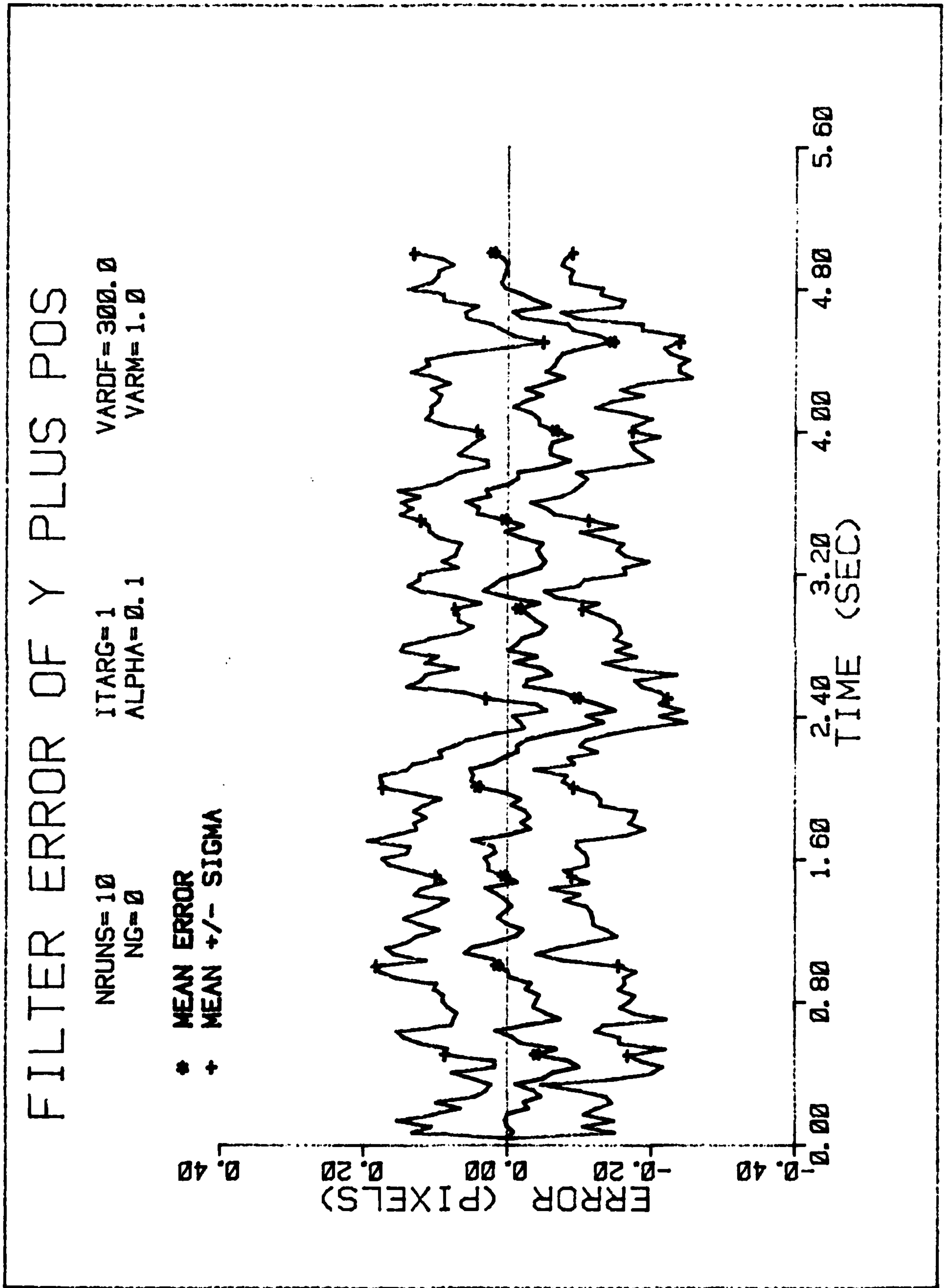


Figure D-66 Case 7 GM Performance Plot

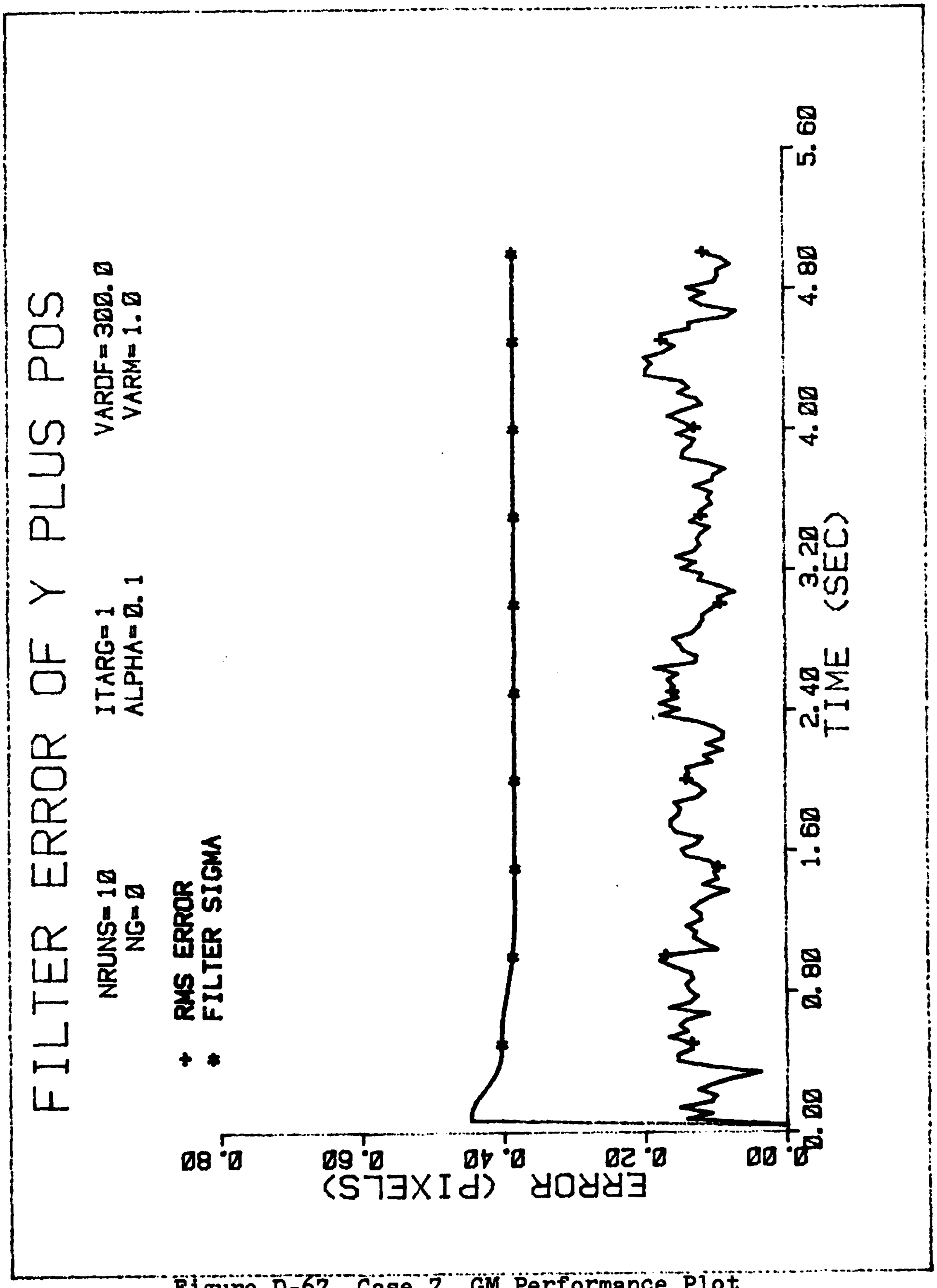


Figure D-67 Case 7 GM Performance Plot

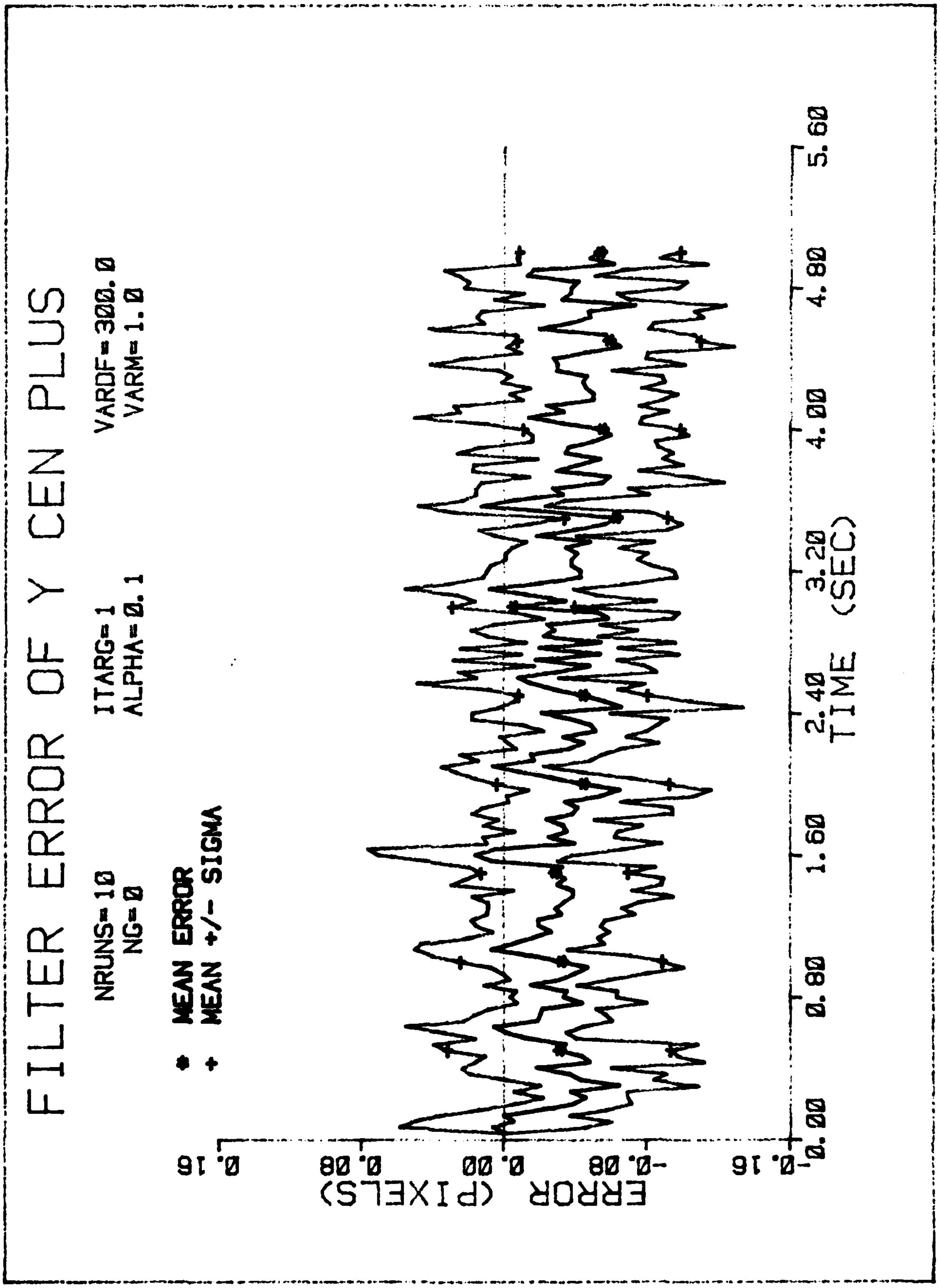


Figure D-68 Case 7 GM Performance Plot

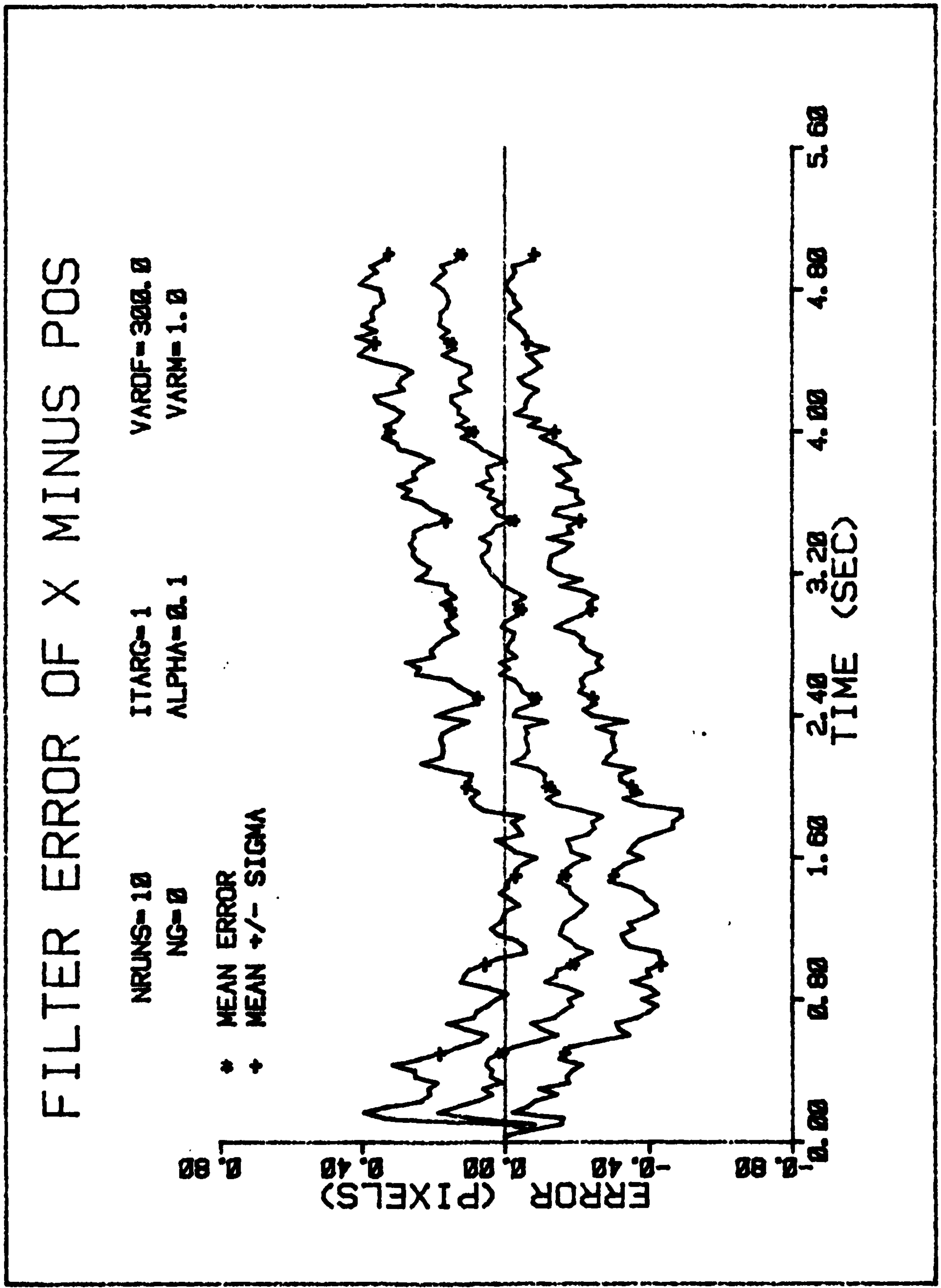


Figure D-69 Case 8 GM Performance Plot

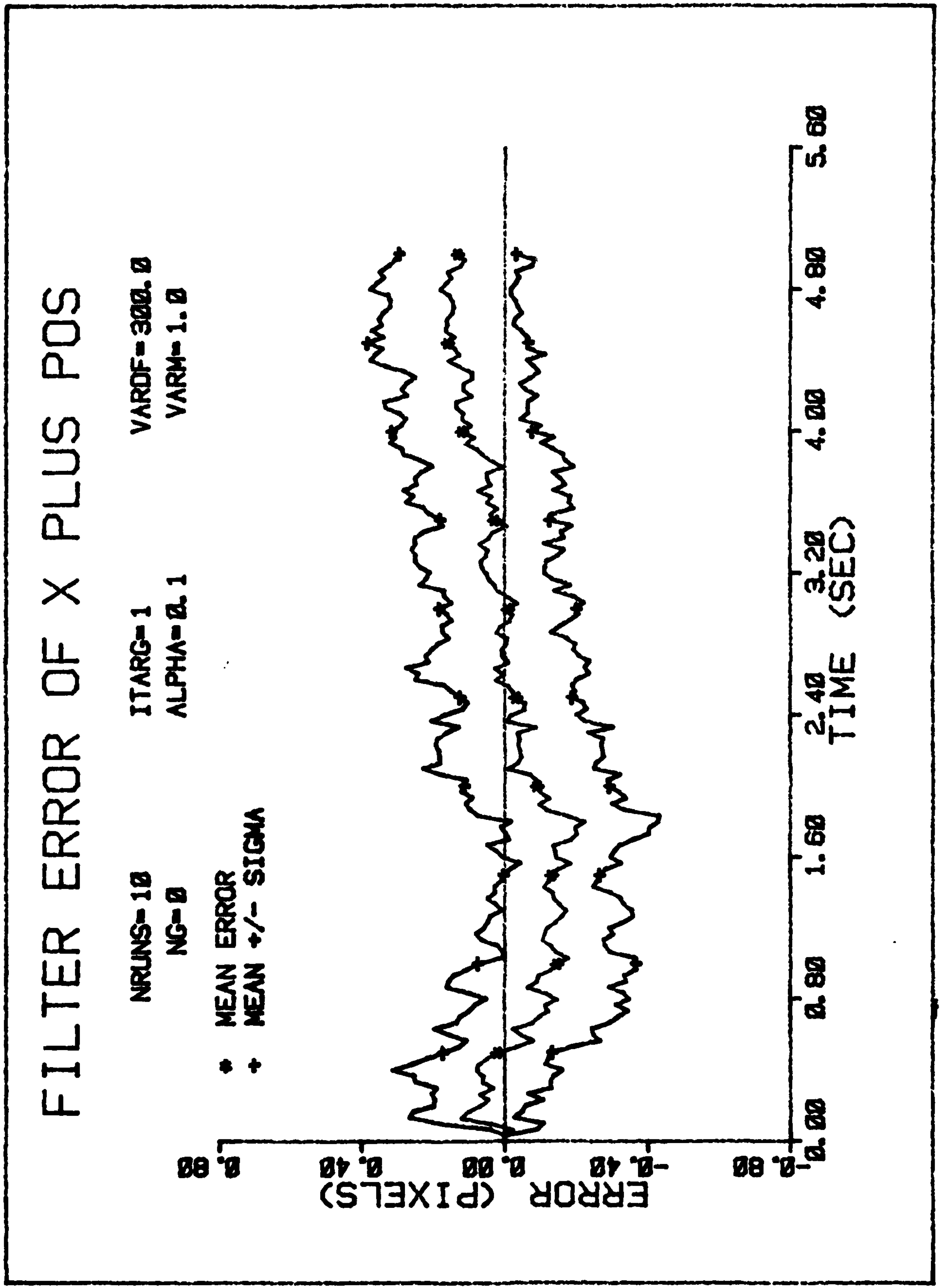


Figure D-70 Case 8 GM Performance Plot

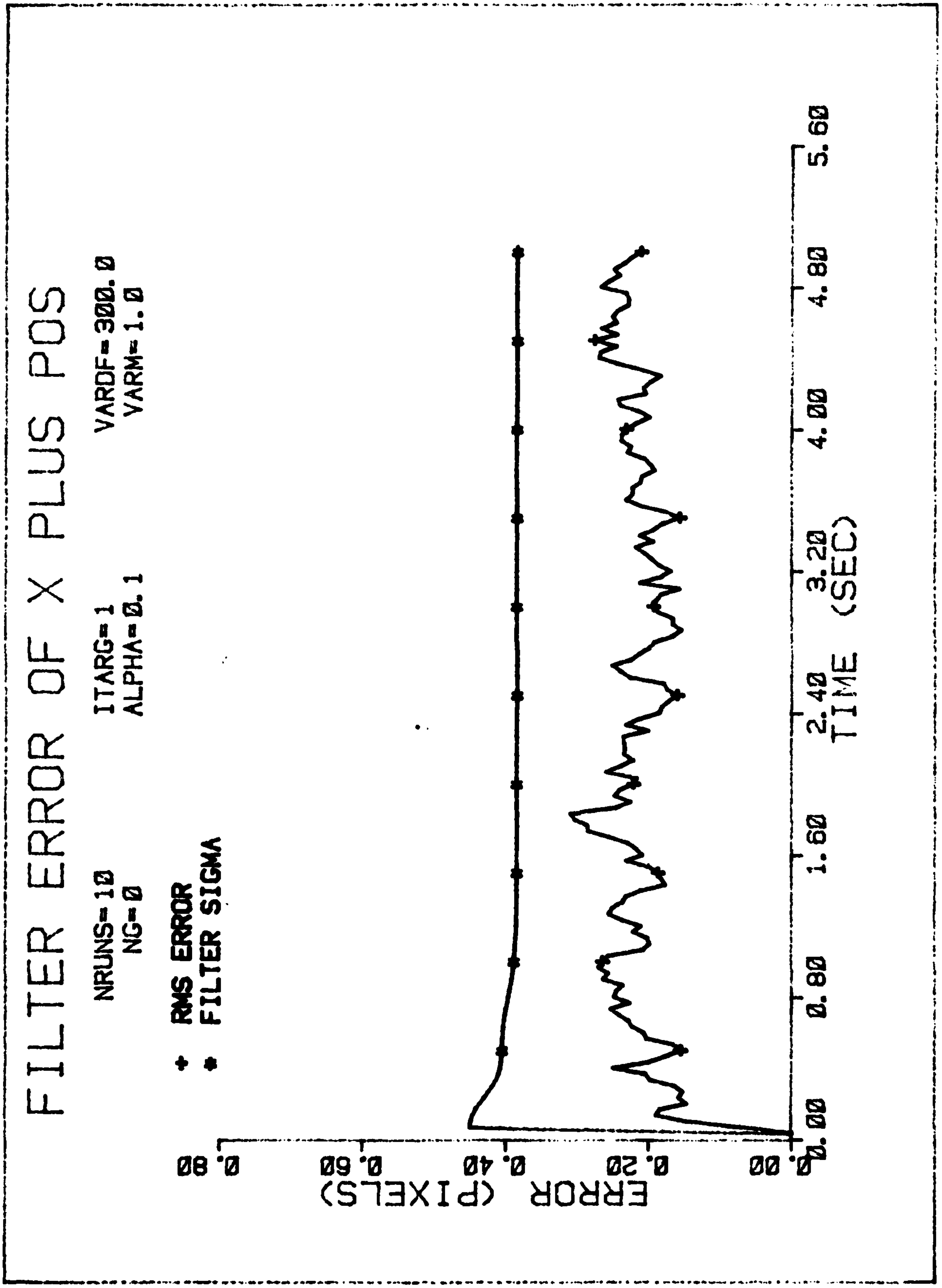


Figure D-71 Case 8 GM Performance Plot

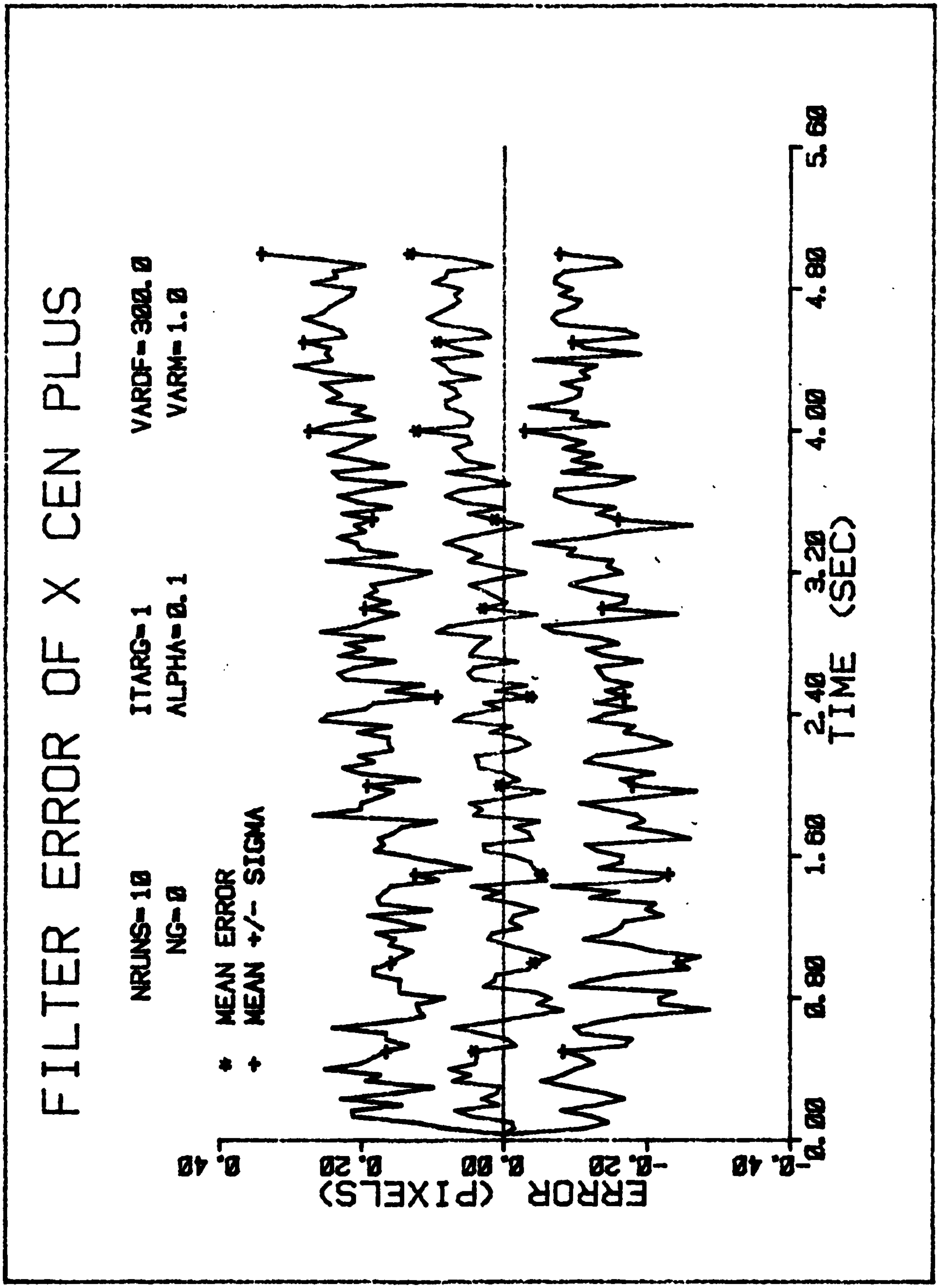


Figure D-72 Case 8 GM Performance Plot

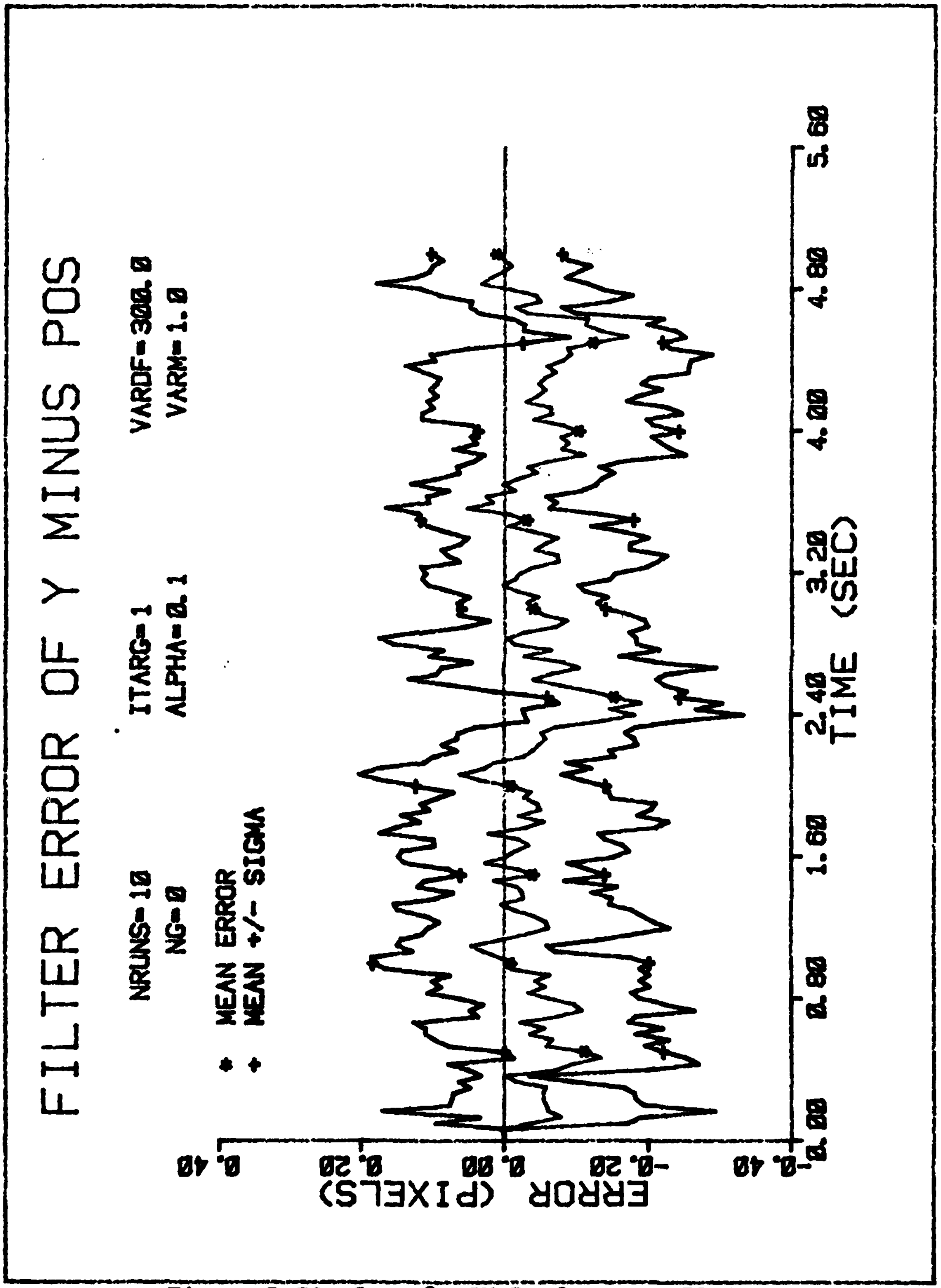


Figure D-73 Case 8 GM Performance Plot

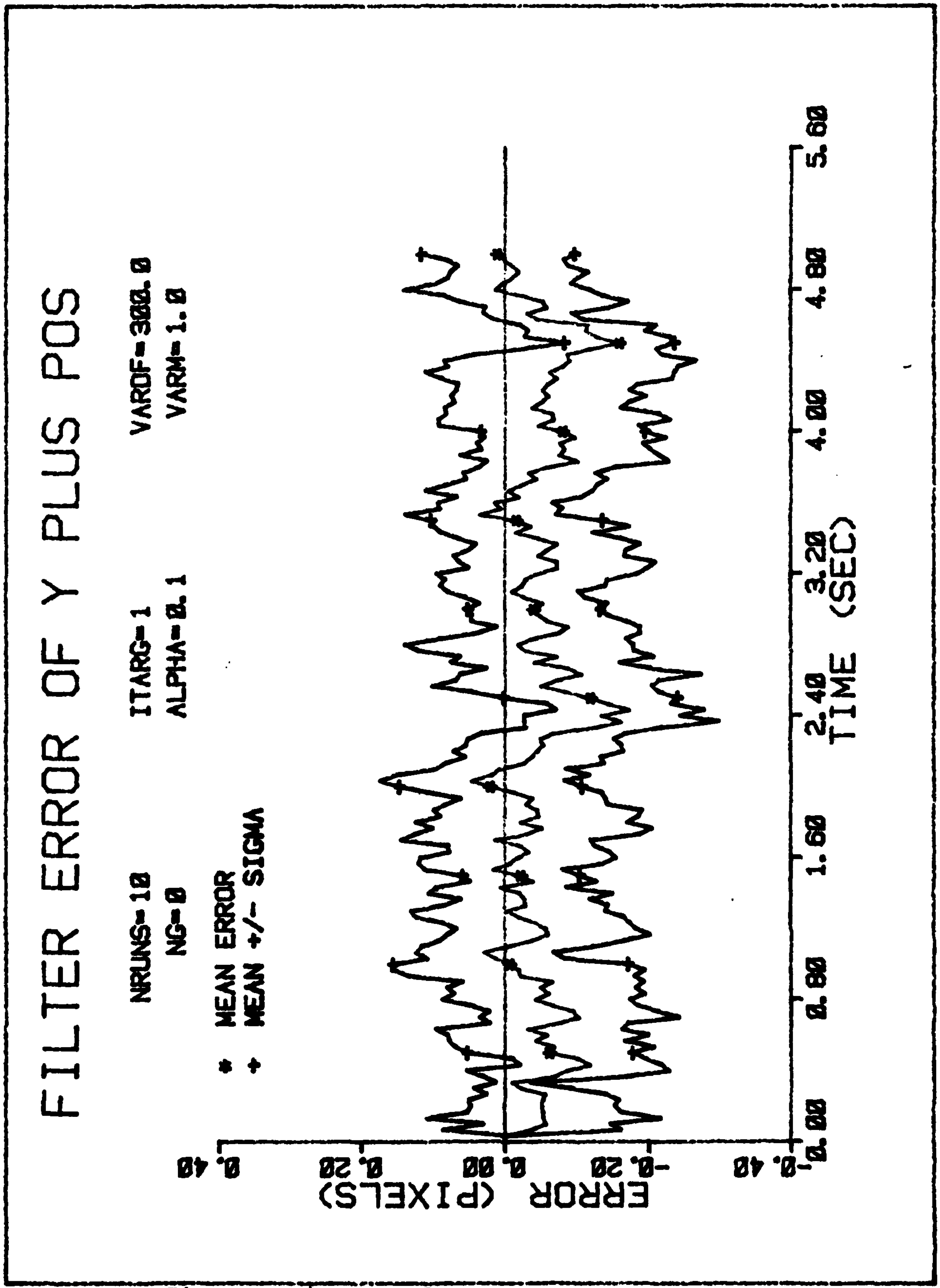


Figure D-74 Case 8 GM Performance Plot

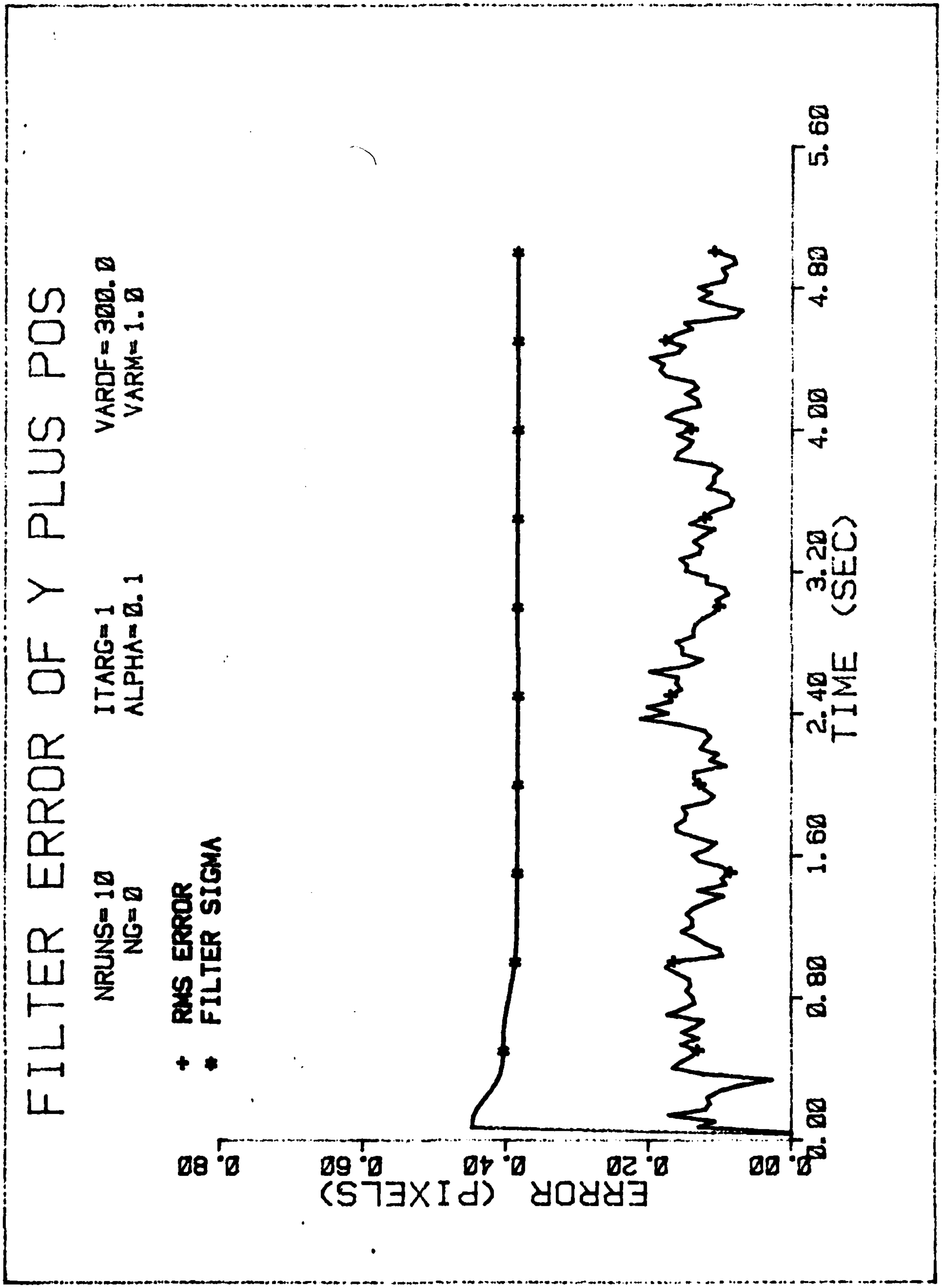


Figure D-75 Case 8 GM Performance Plot

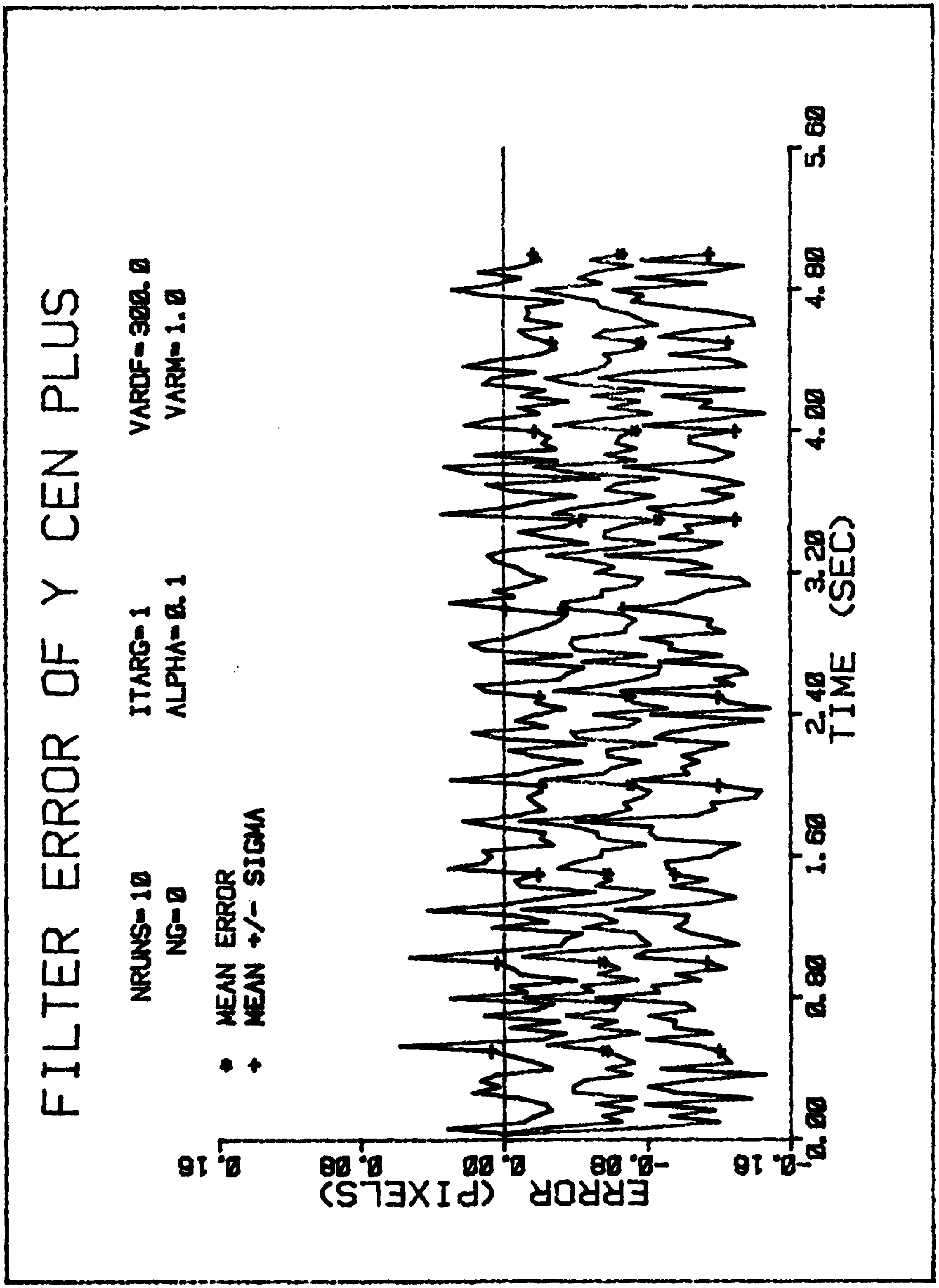


Figure D-76 Case 8 GM Performance Plot

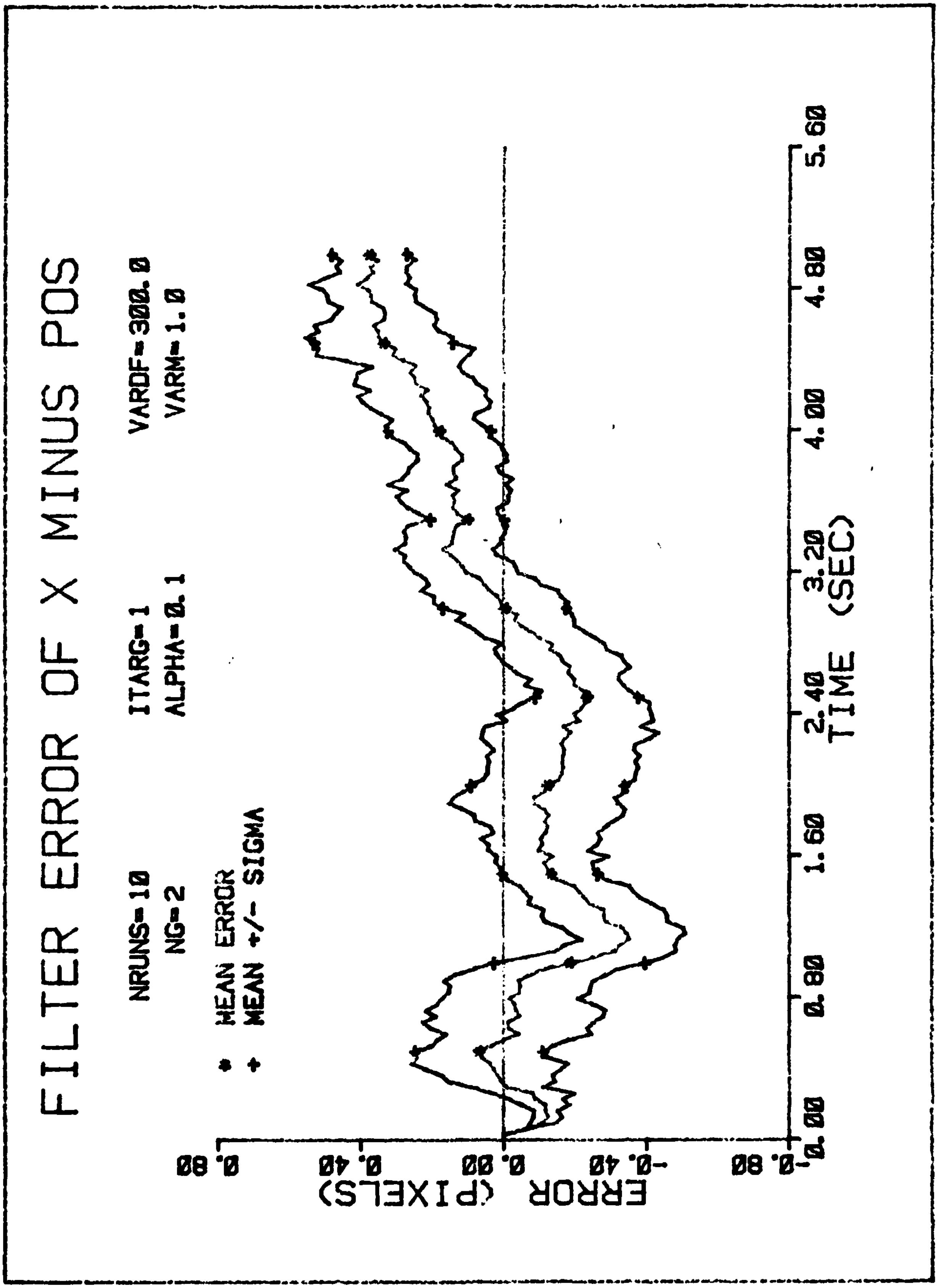


Figure D-77 Case 9 GM Performance Plot

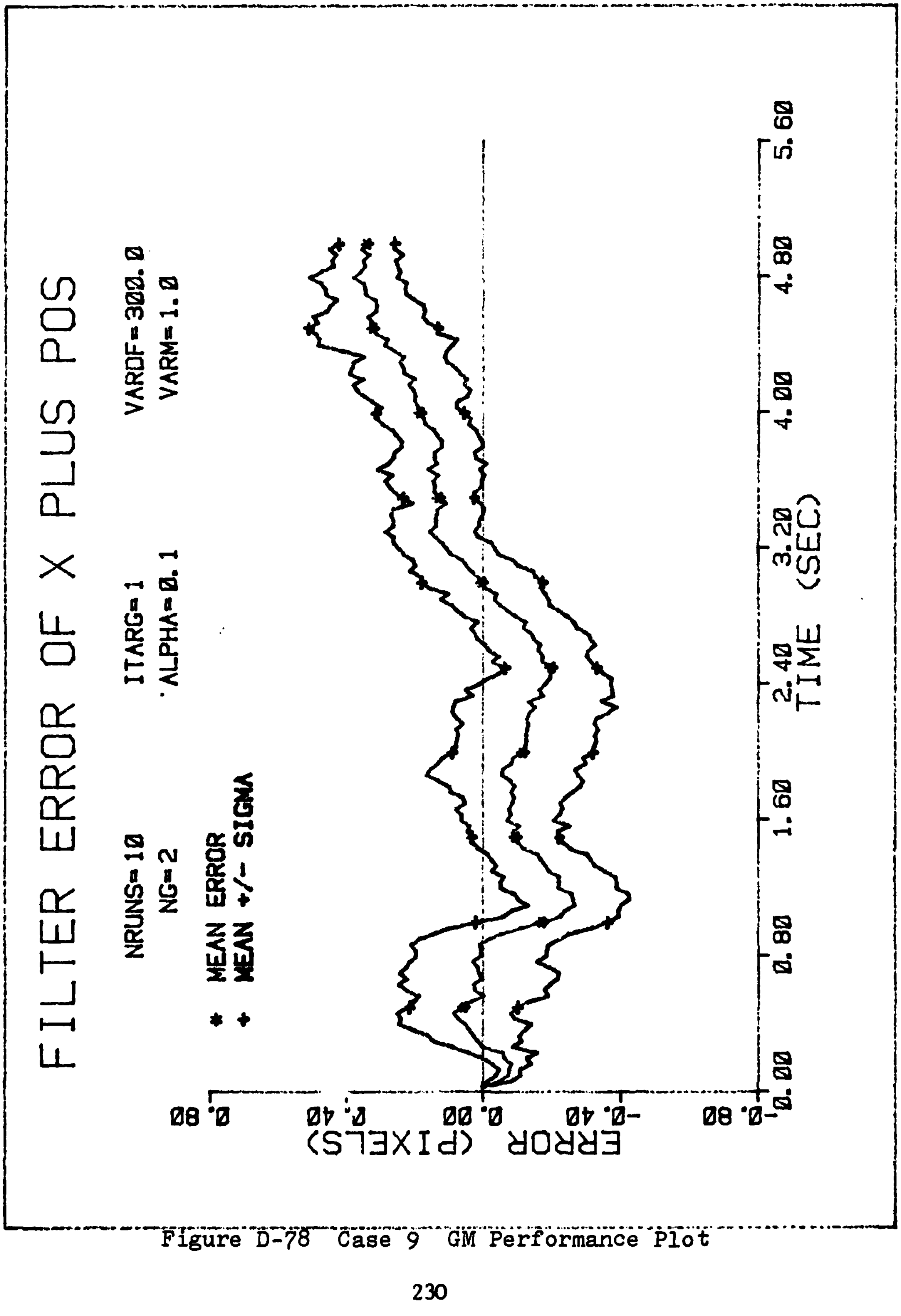


Figure D-78 Case 9 GM Performance Plot

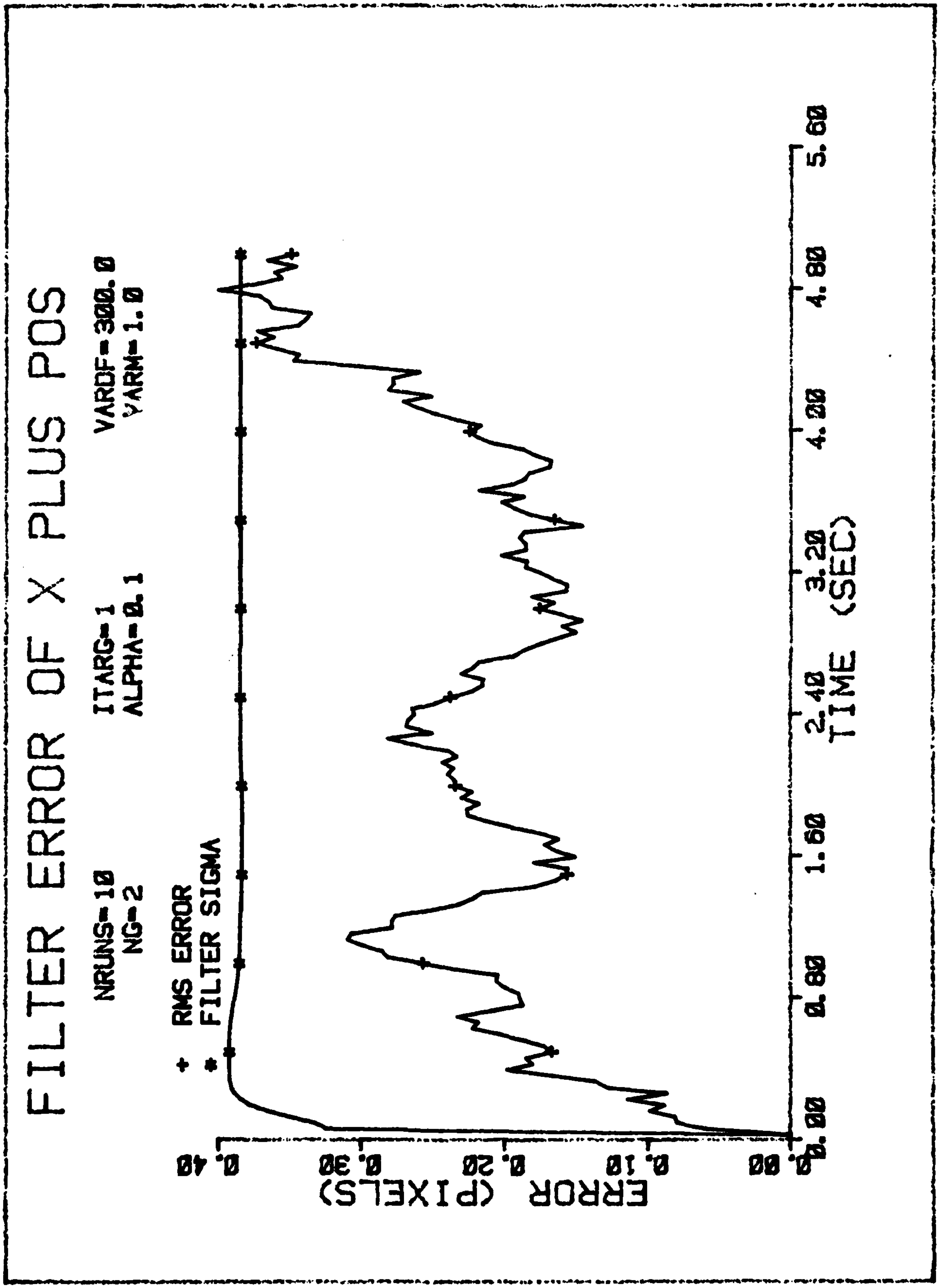


Figure D-79 Case 9 GM Performance Plot

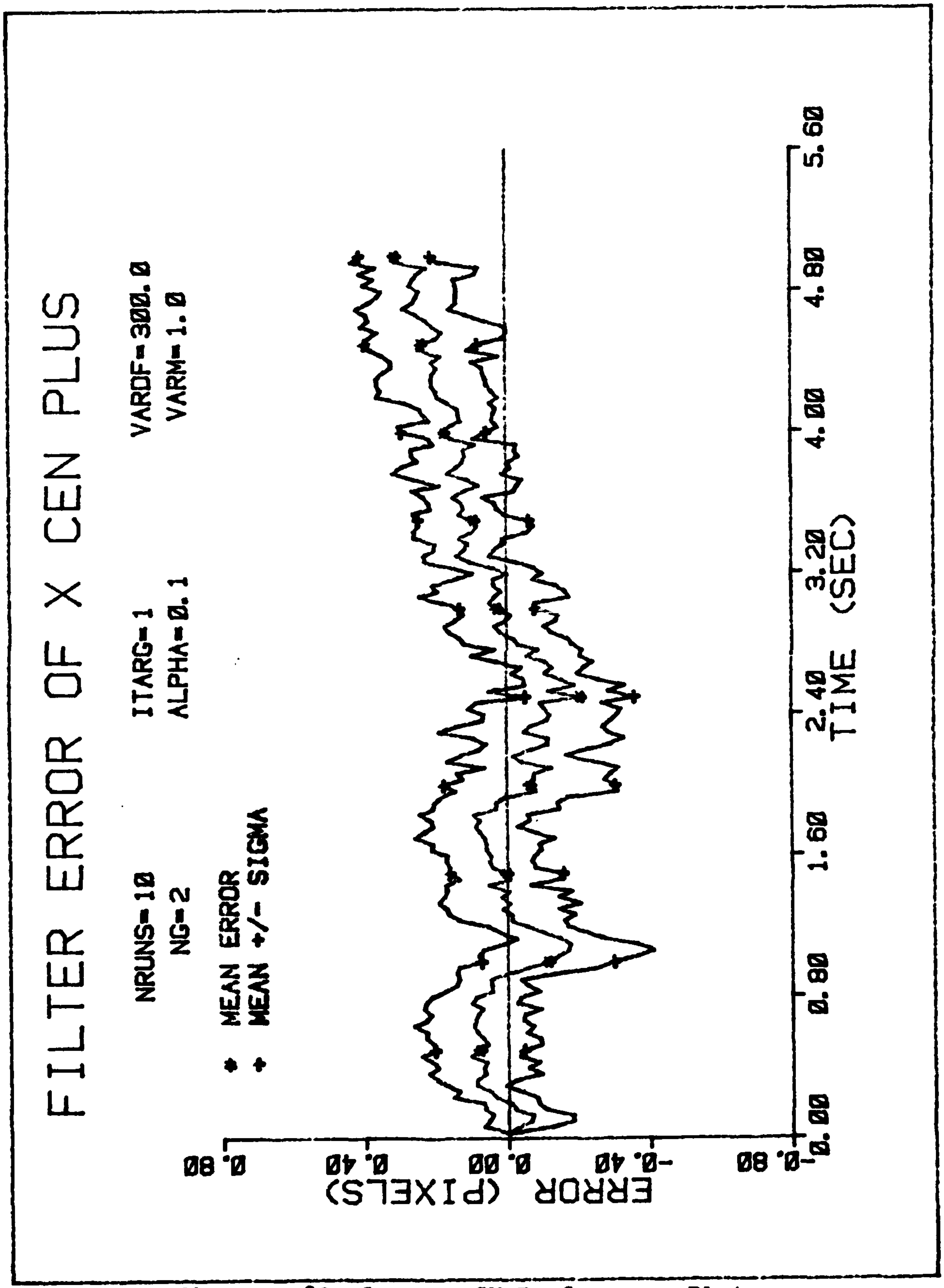


Figure D-80 Case 9 GM Performance Plot

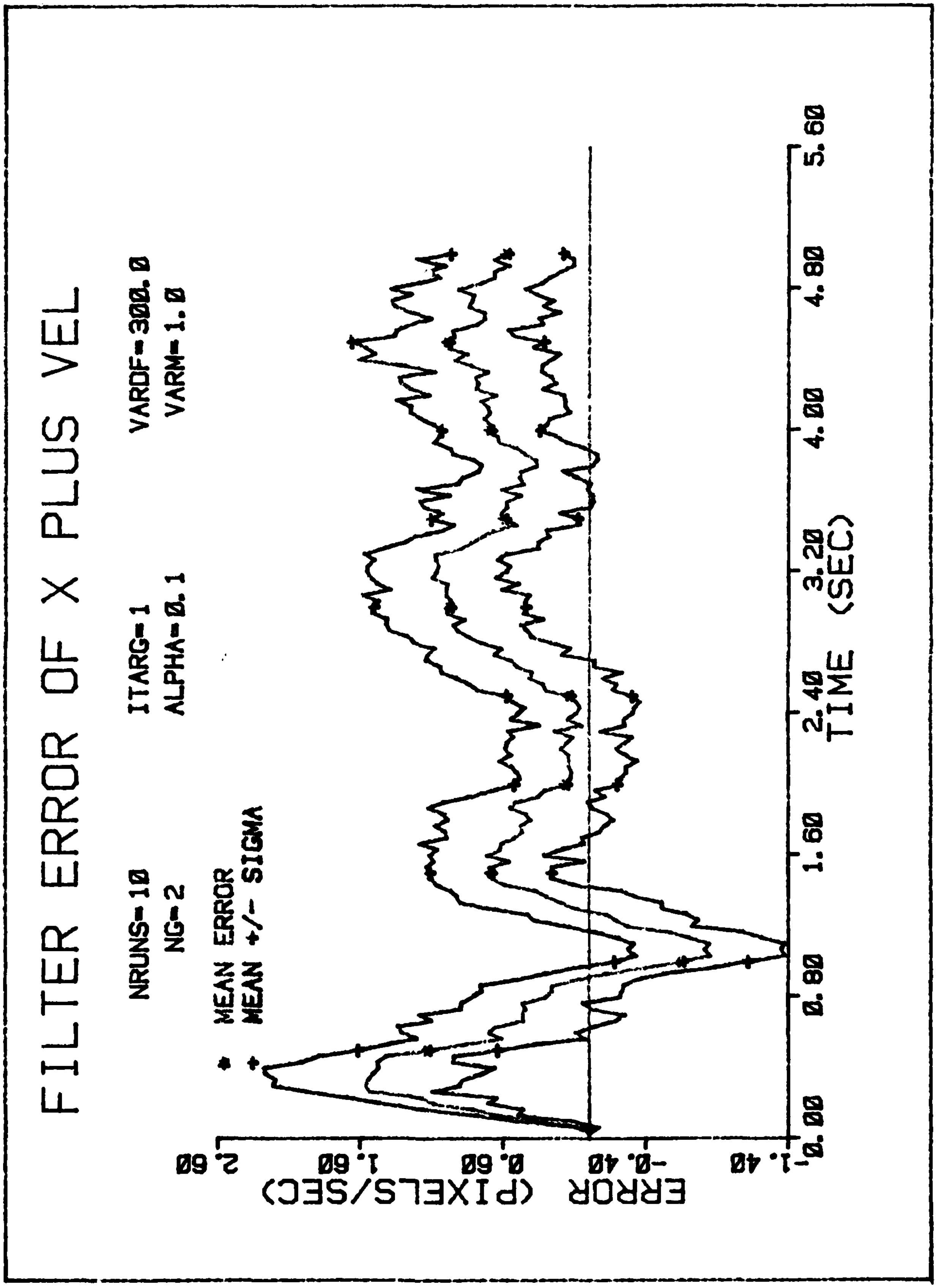


Figure D-81 Case 9 GM Performance Plot

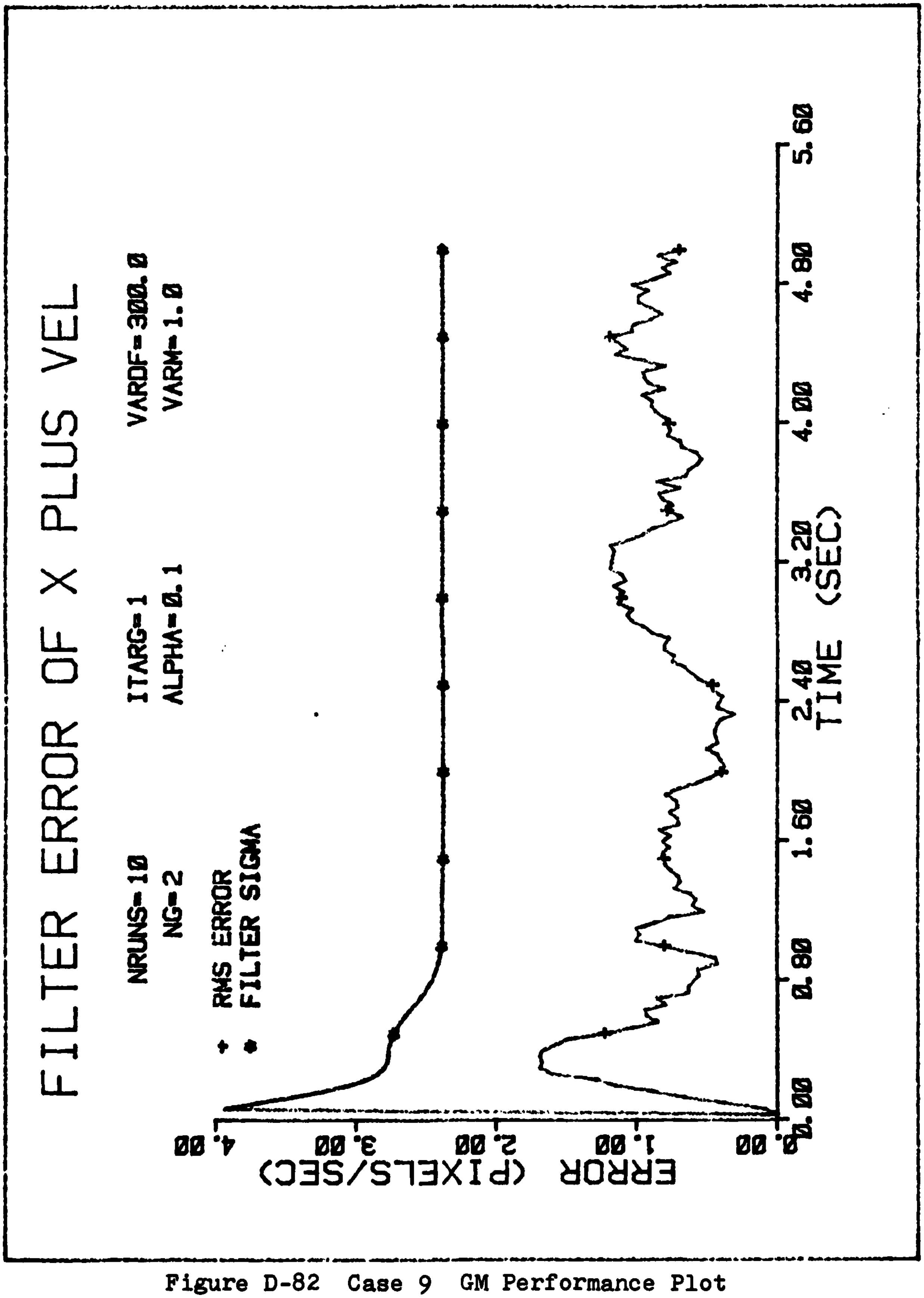


Figure D-82 Case 9 GM Performance Plot

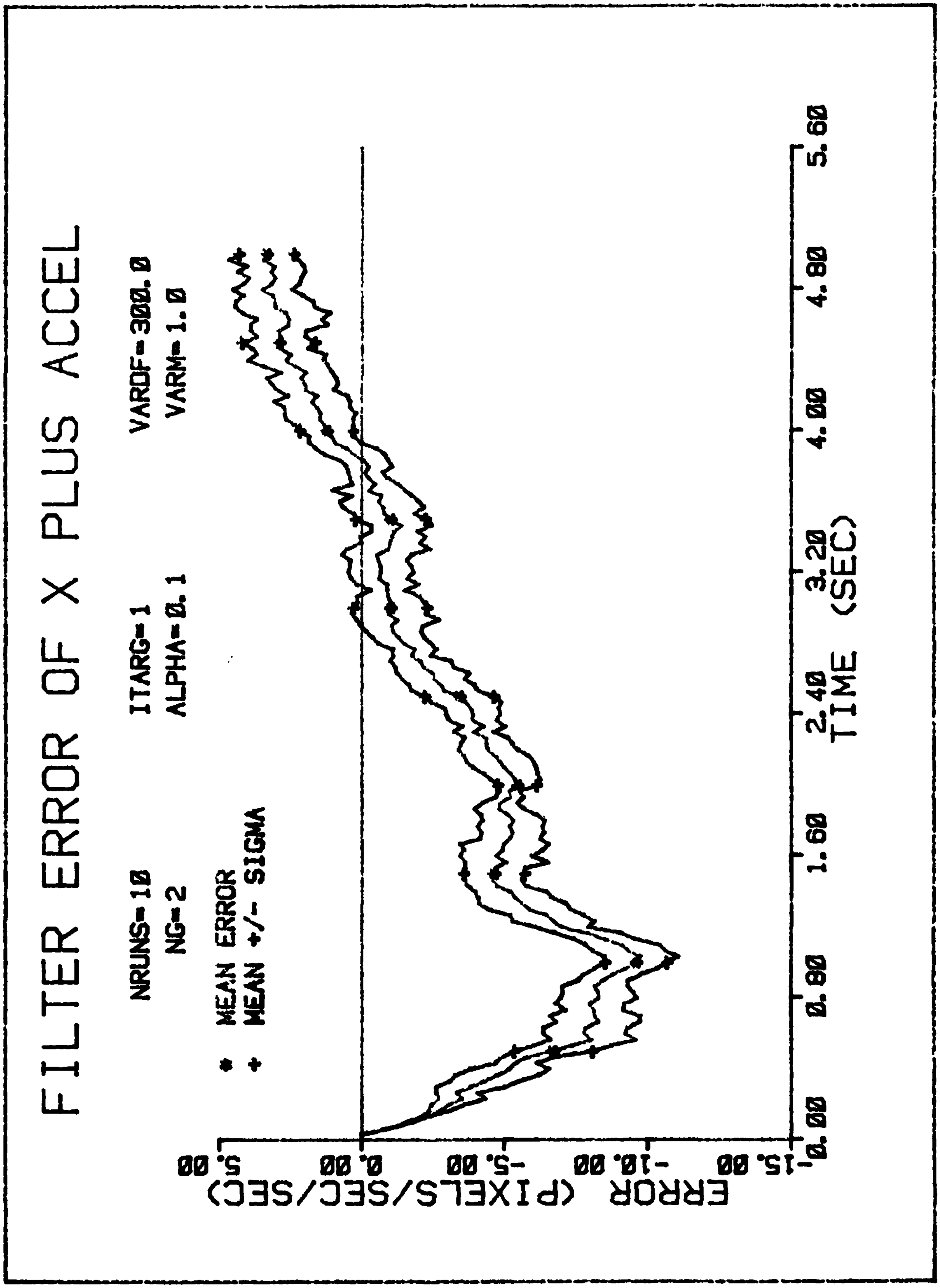


Figure D-83 Case 9 GM Performance Plot

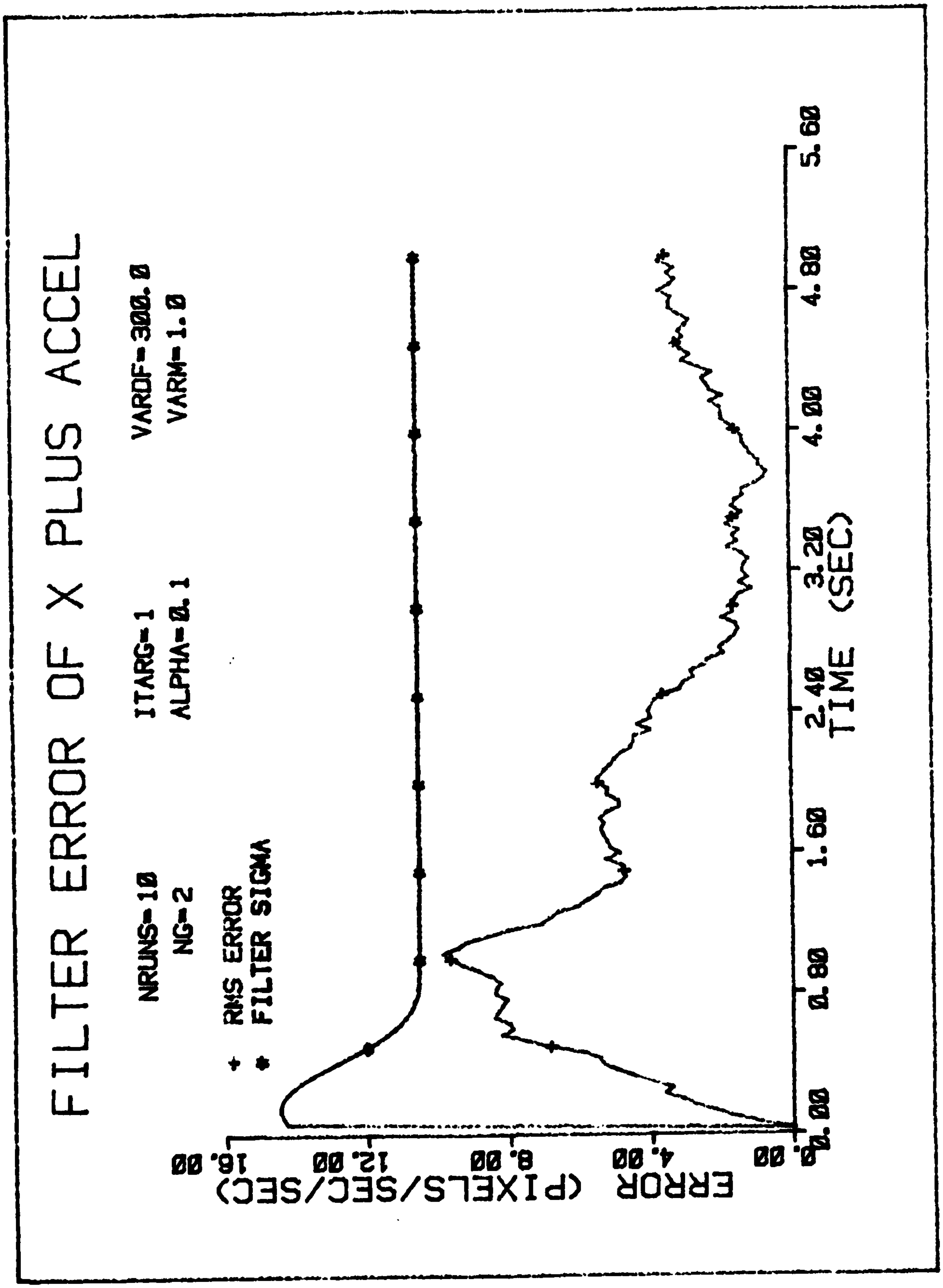


Figure D-84 Case 9 GM Performance Plot

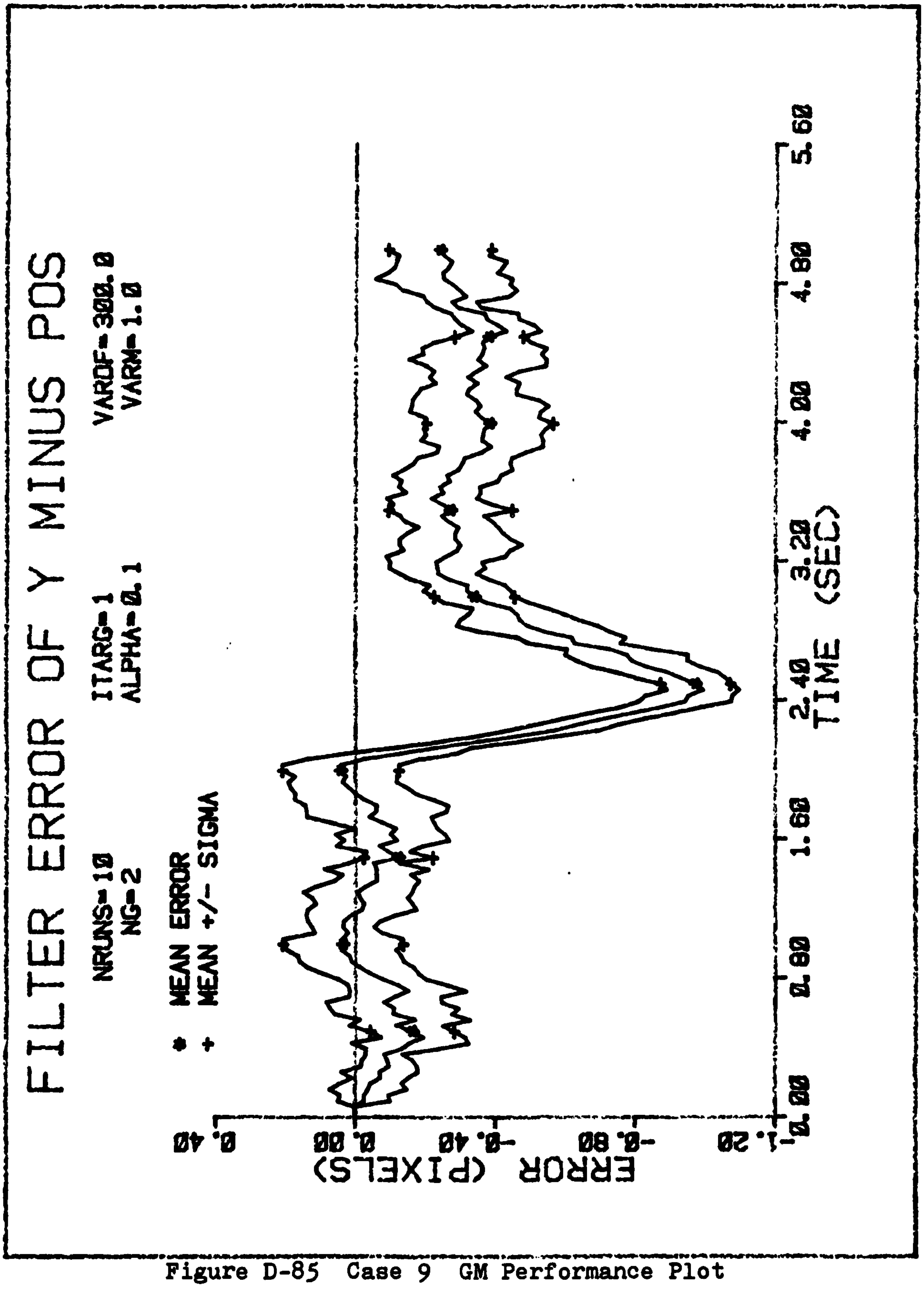


Figure D-85 Case 9 GM Performance Plot

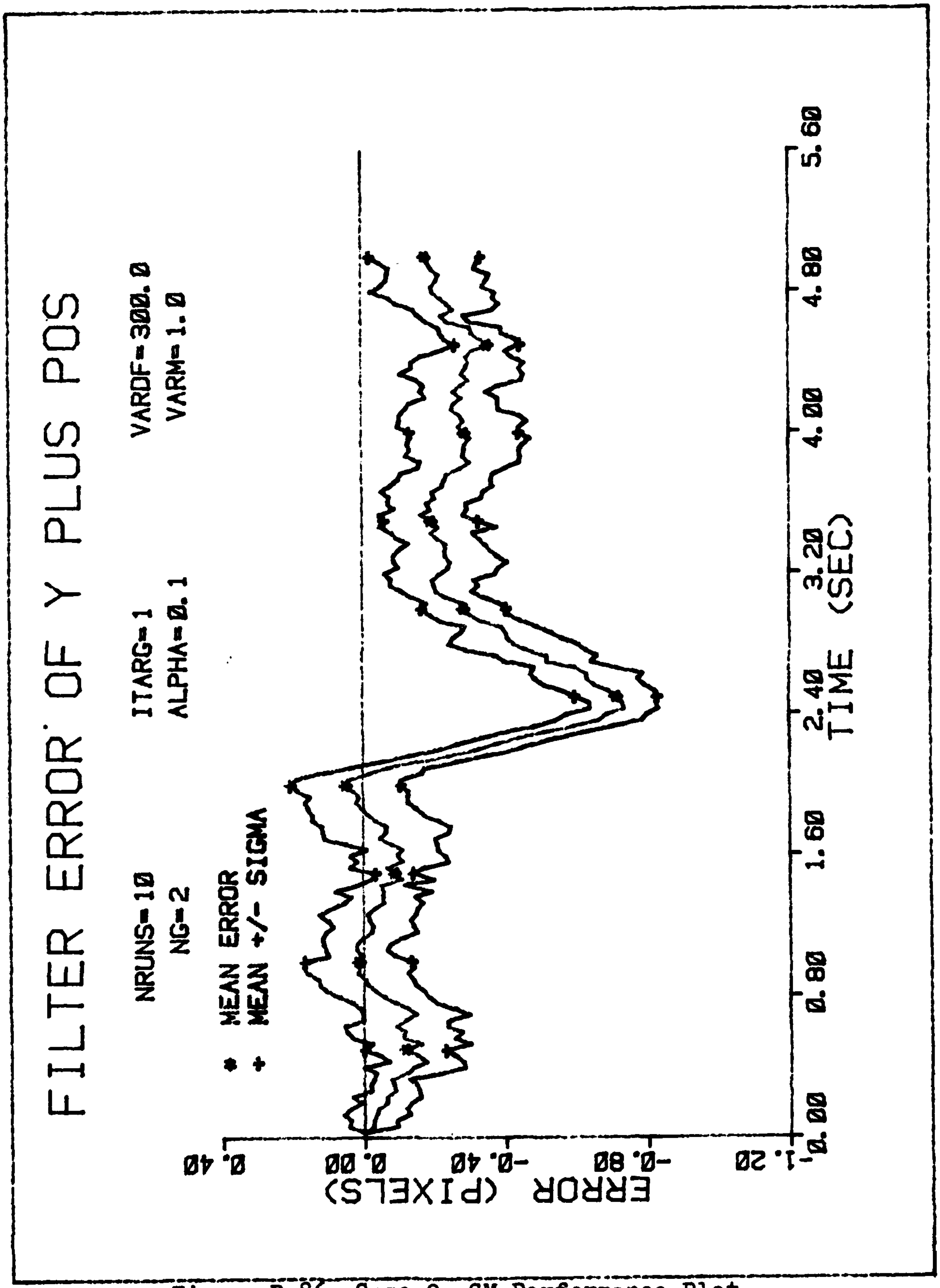


Figure D-86 Case 9 GM Performance Plot

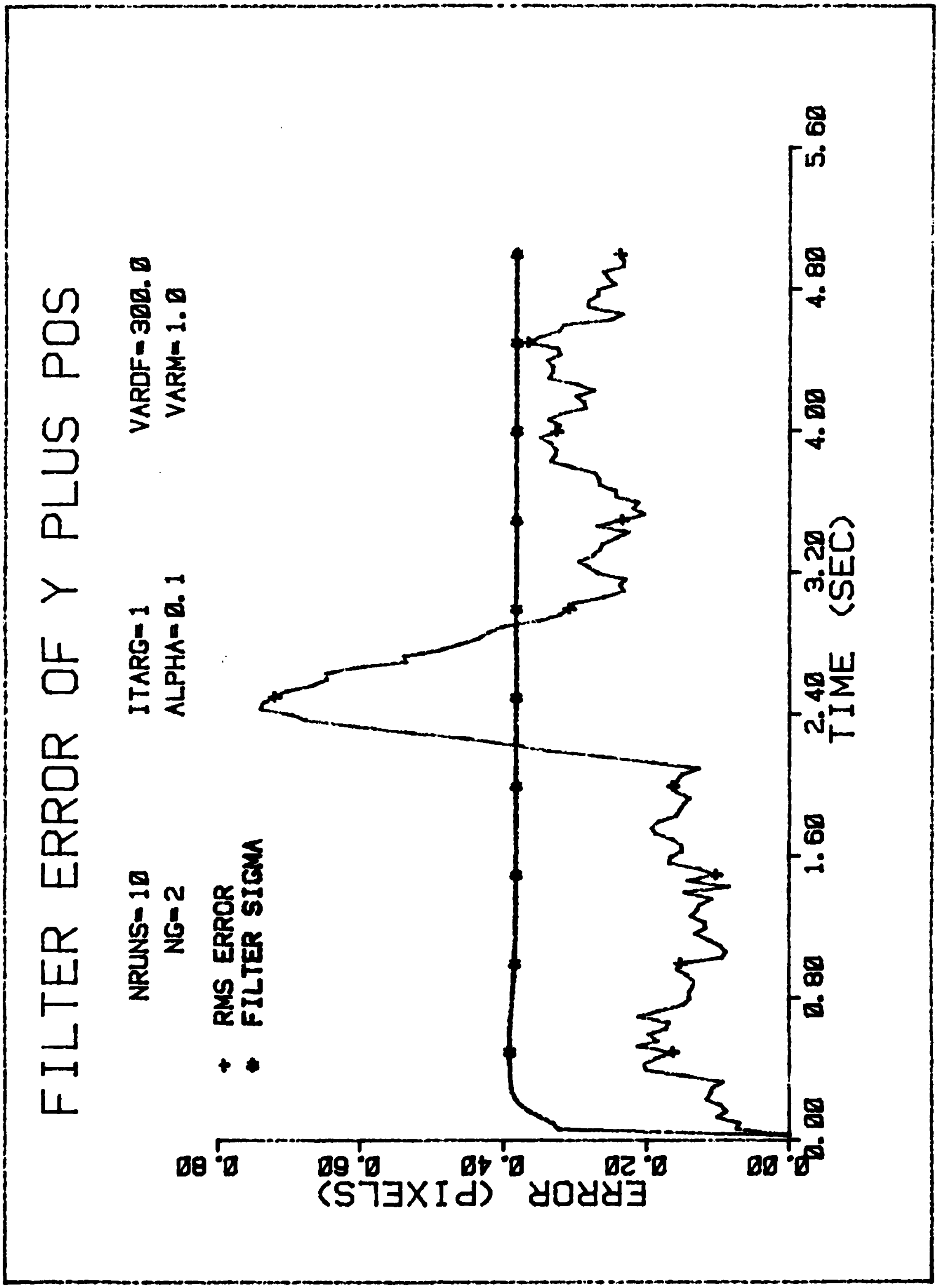


Figure D-87 Case 9 GM Performance Plot

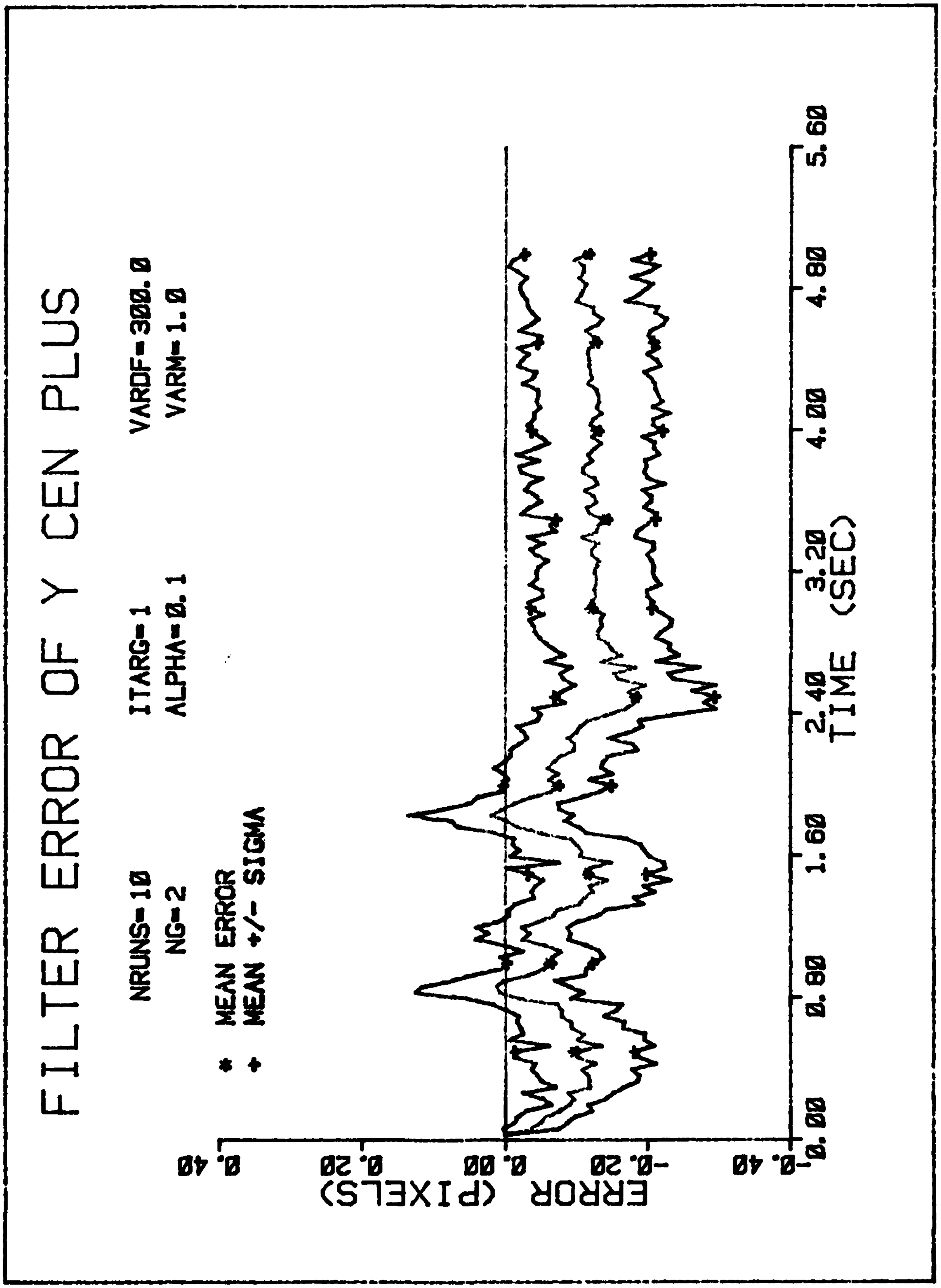


Figure D-88 Case 9 GM Performance Plot

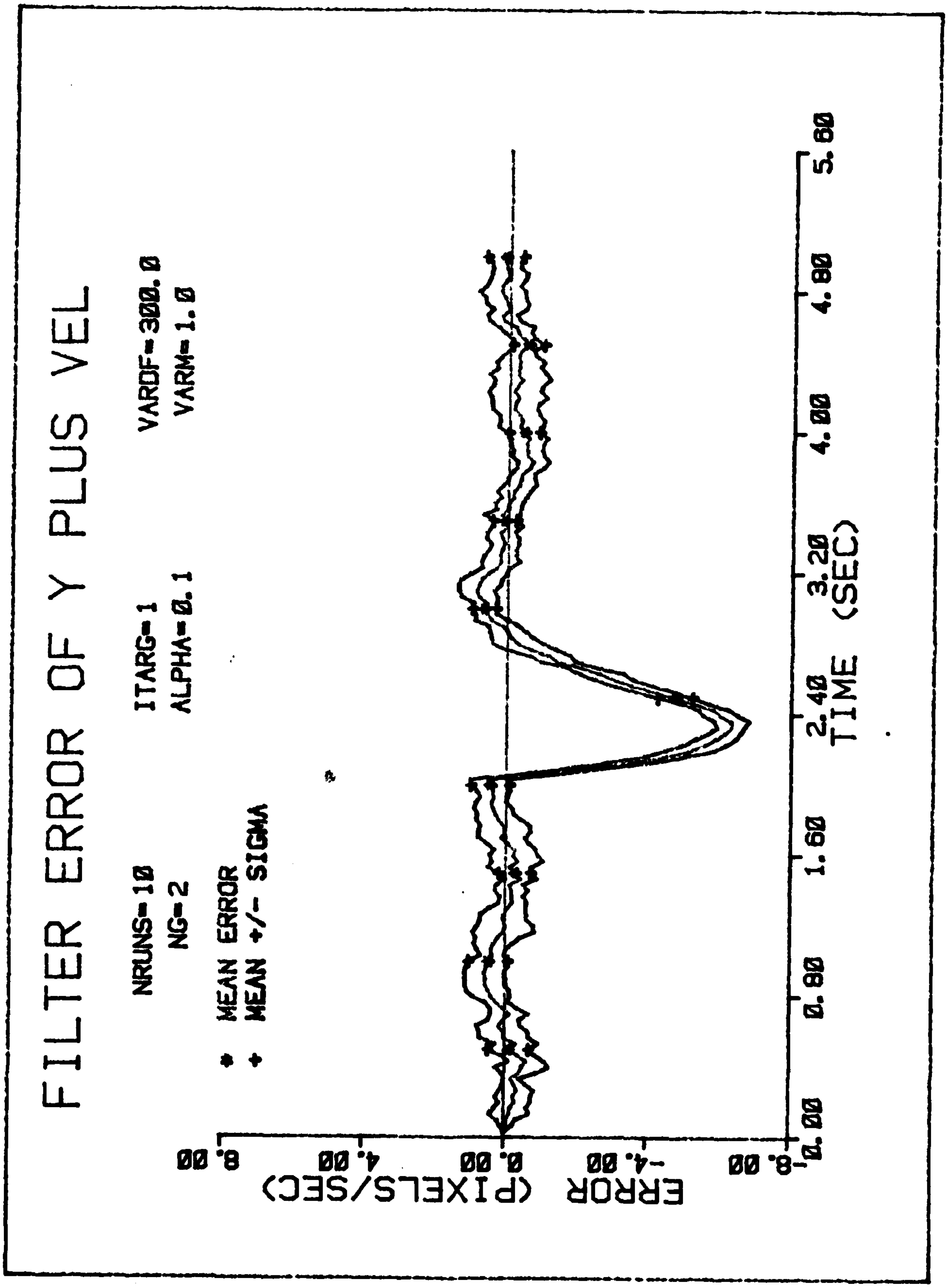


Figure D-89 Case 9 GM Performance Plot

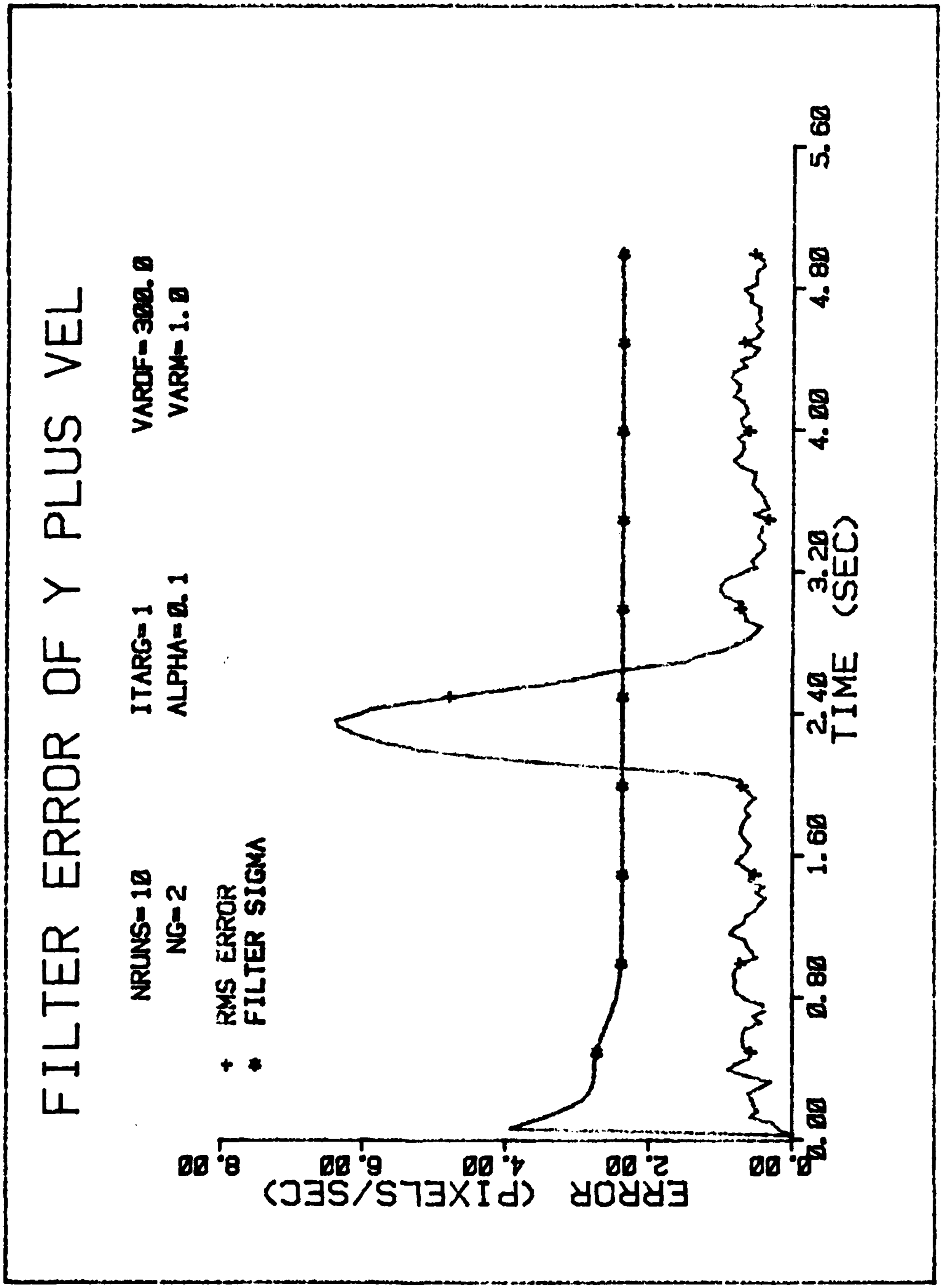


Figure D-90 Case 9 GM Performance Plot

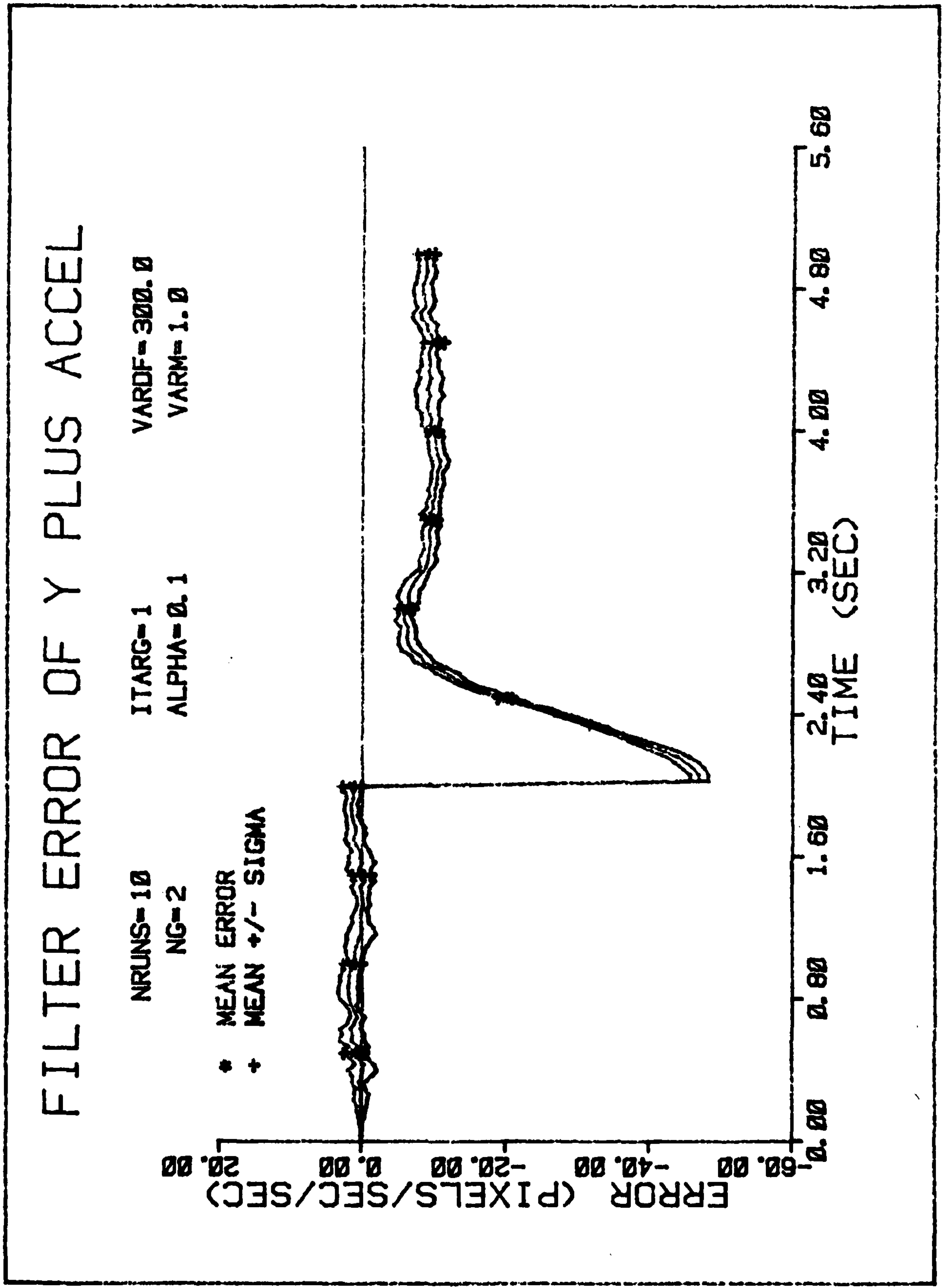


Figure D-91 Case 9 GM Performance Plot

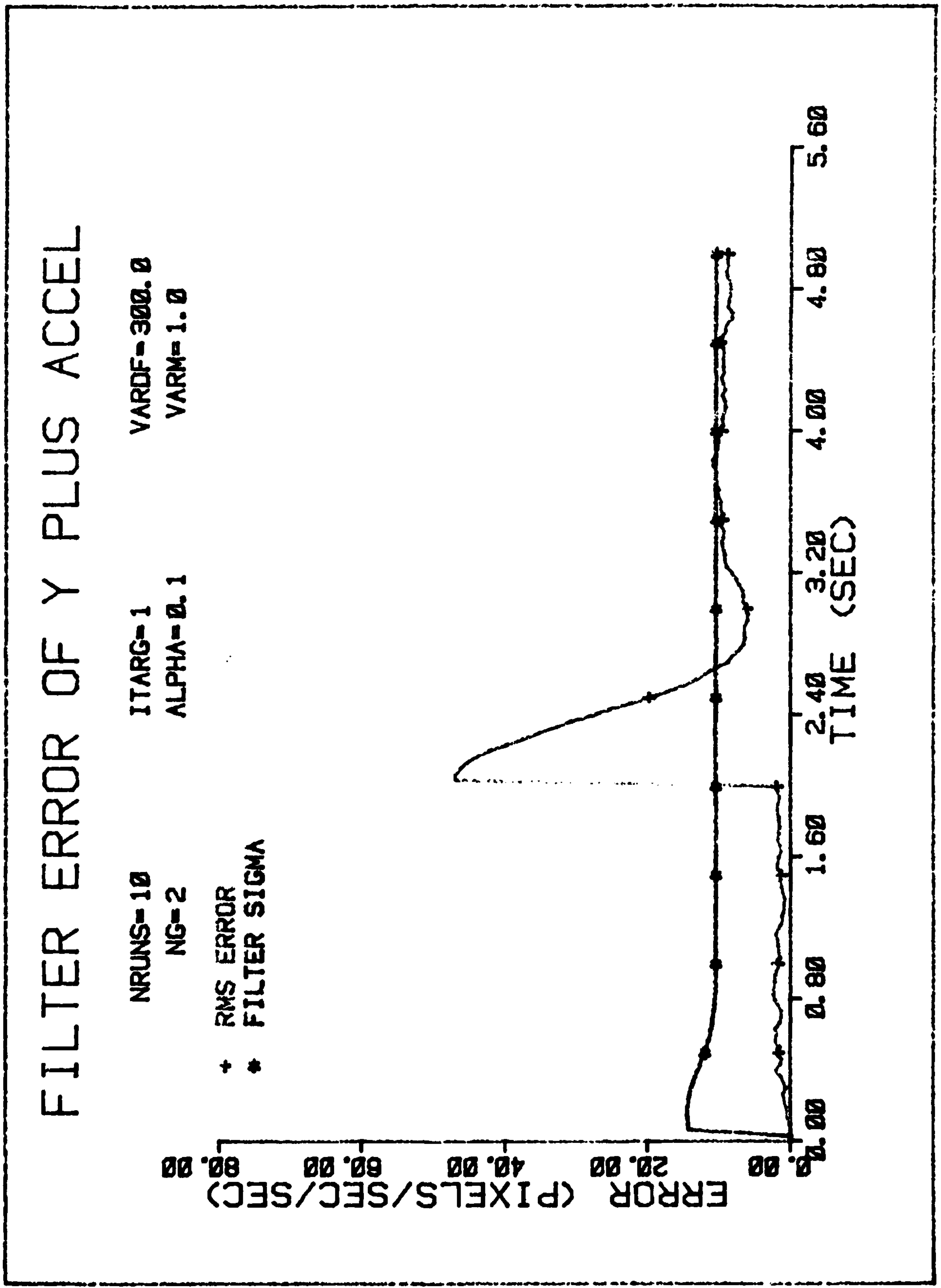


Figure D-92 Case 9 GM Performance Plot

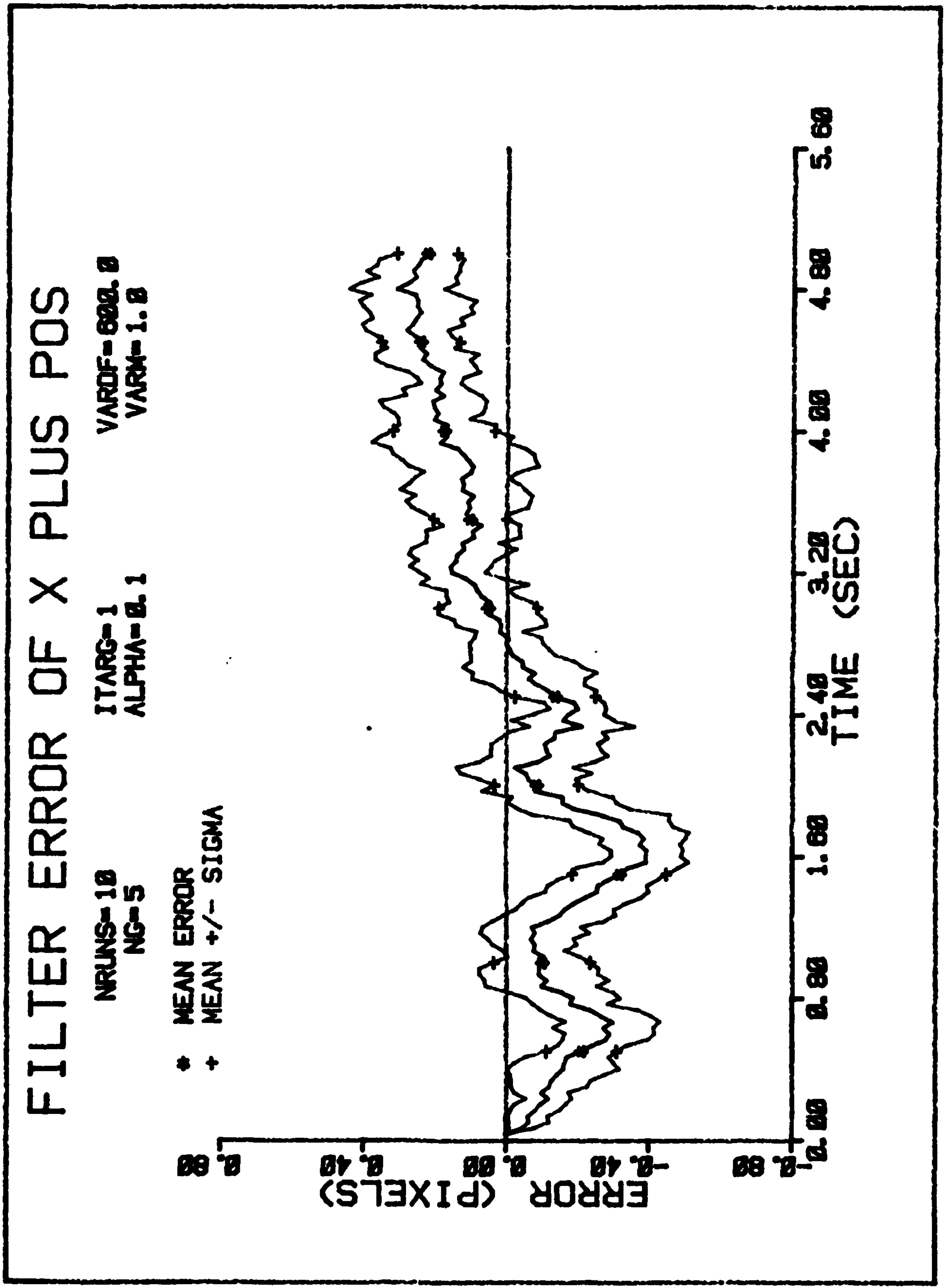


Figure D-93 Case 10 GM Performance Plot

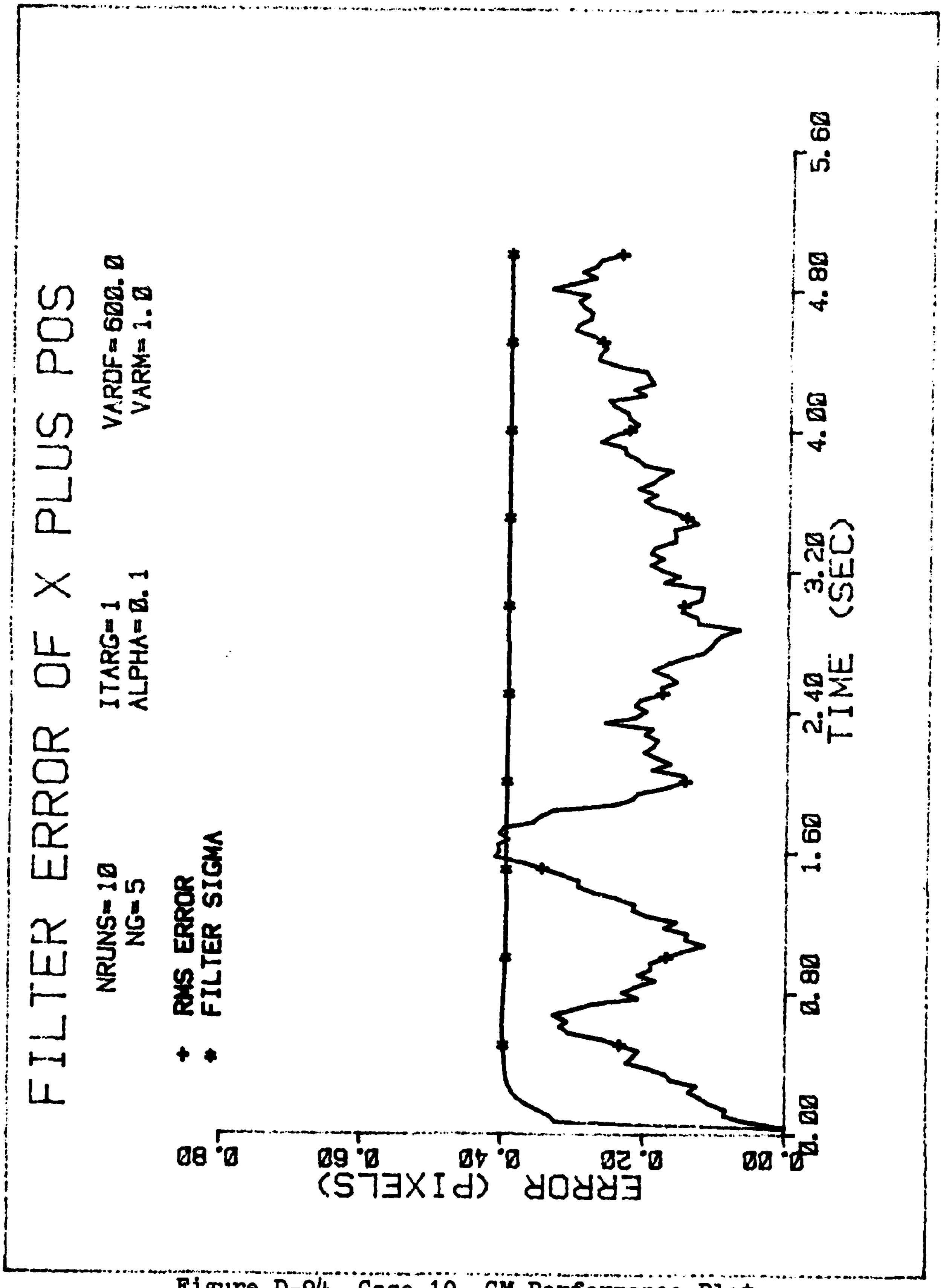


Figure D-94 Case 10 GM Performance Plot

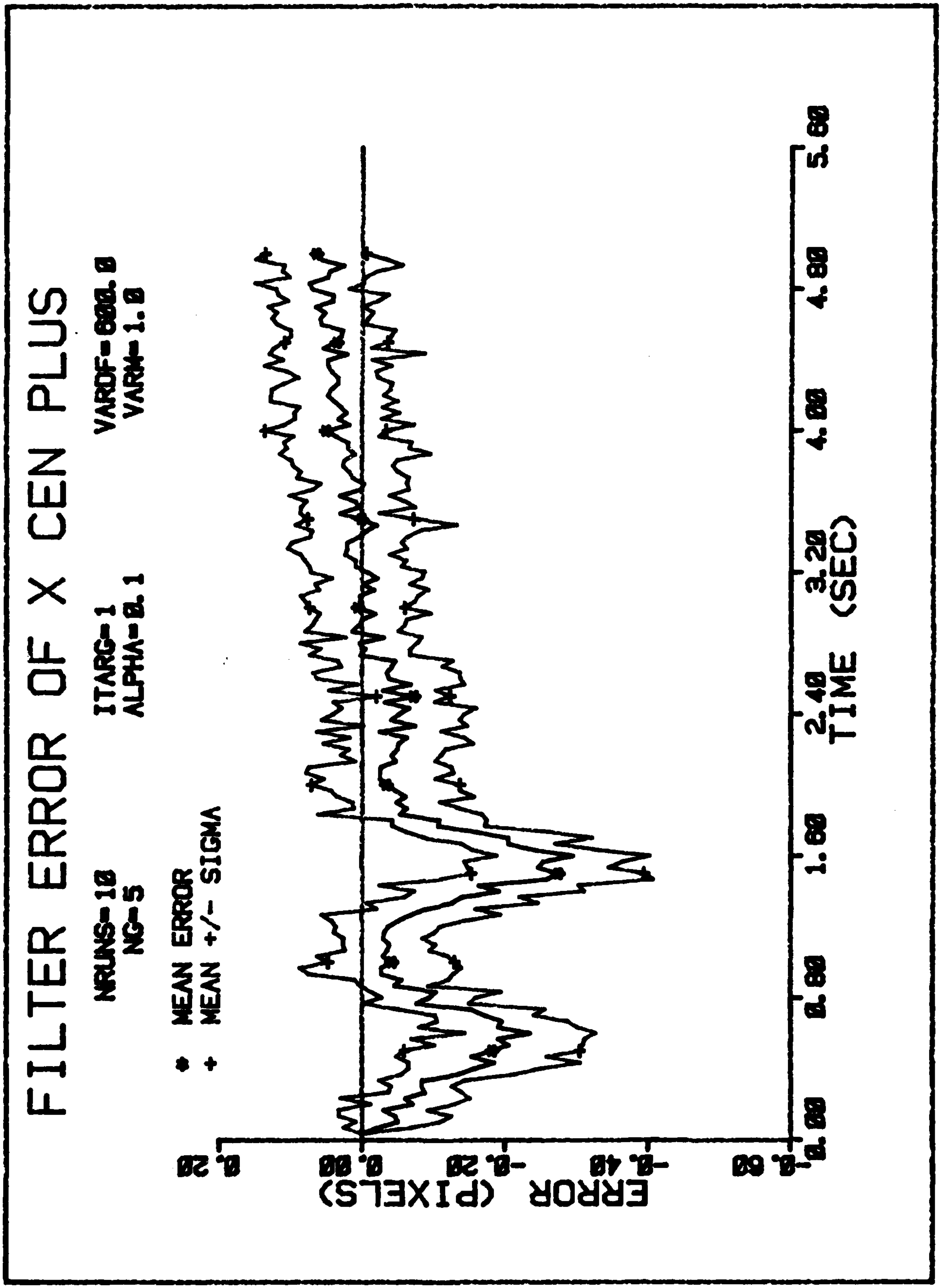


Figure D-95 Case 10 GM Performance Plot

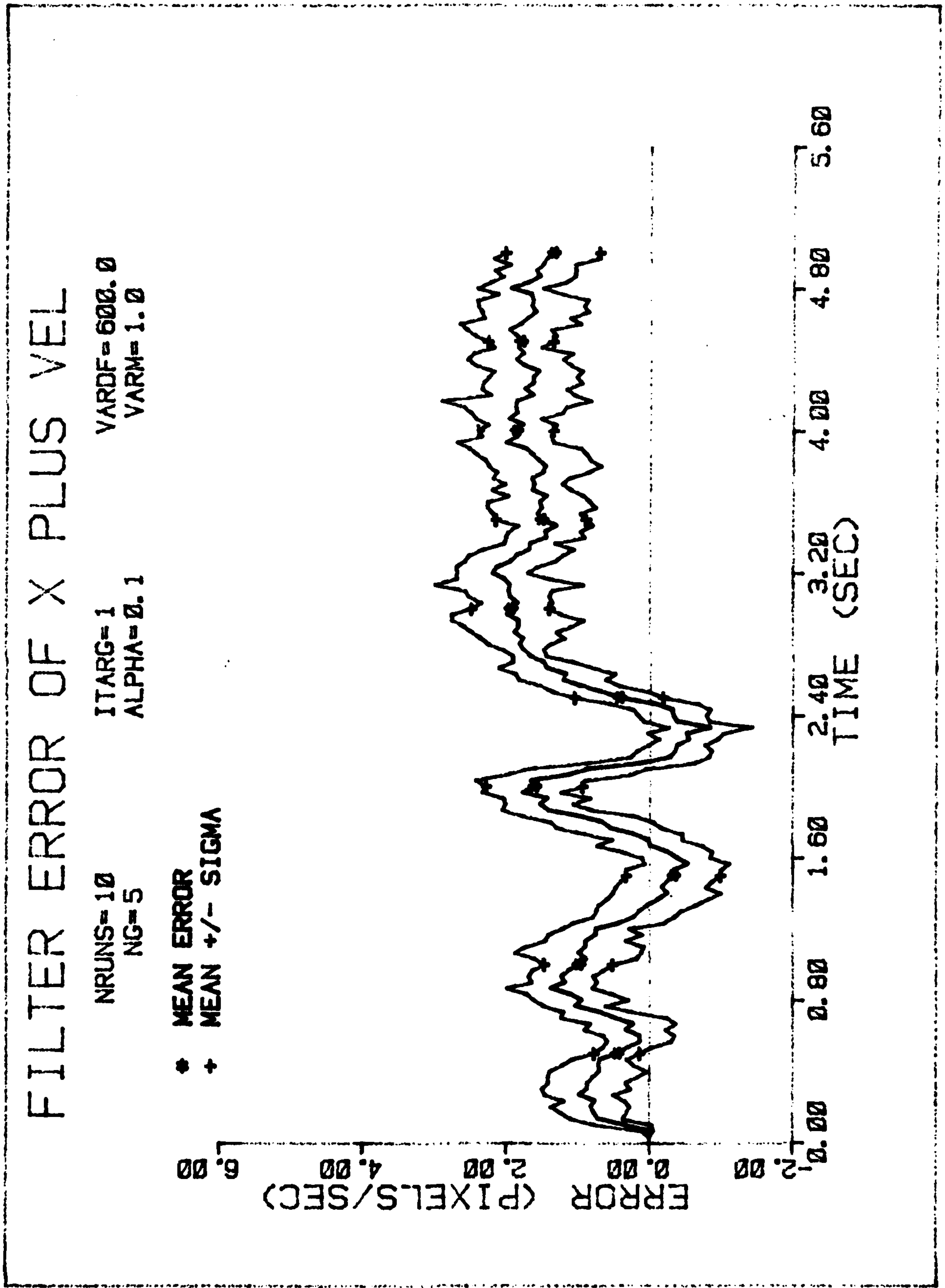


Figure D-96 Case 10 GM Performance Plot

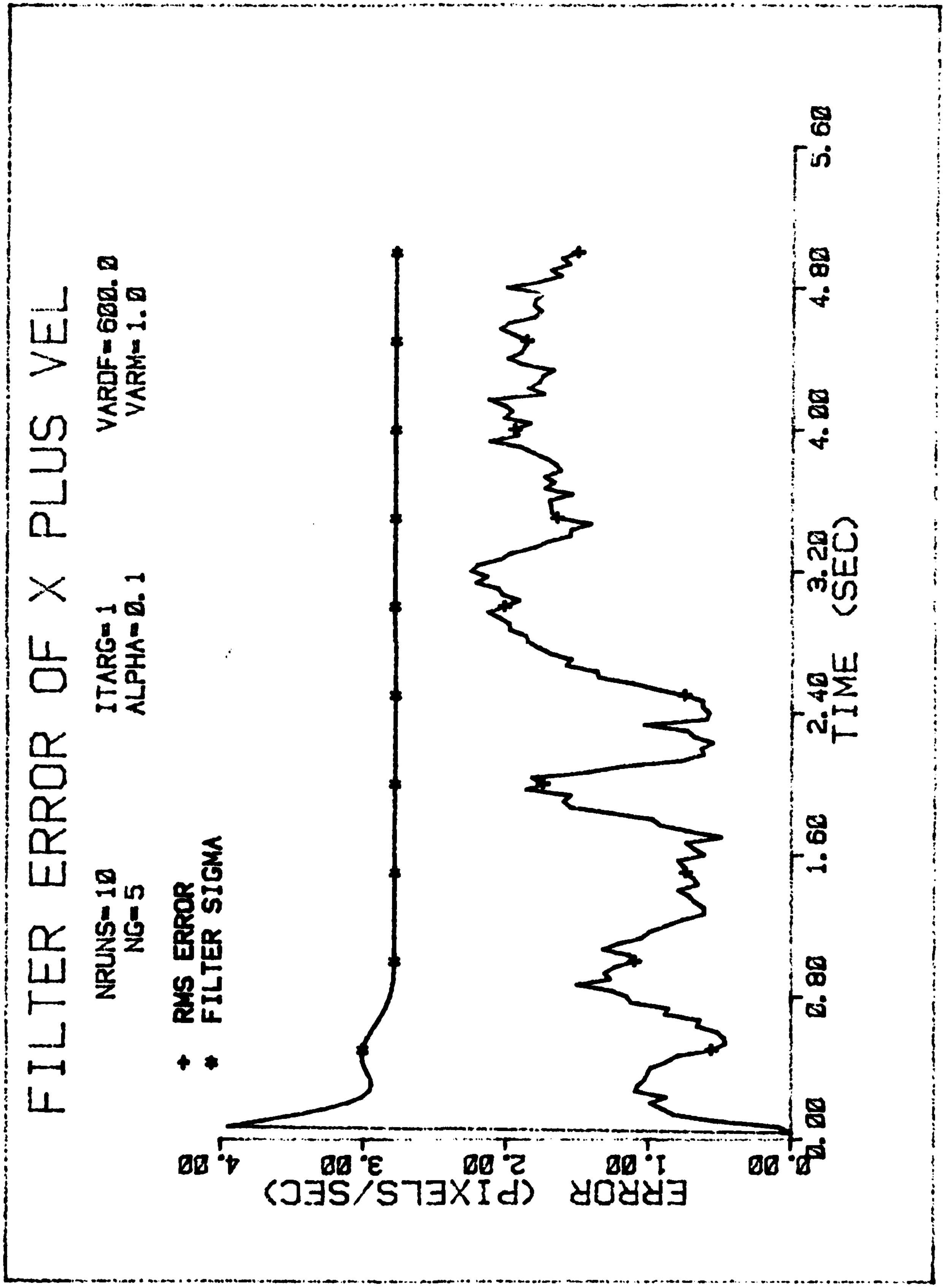


Figure D-97 Case 10 GM Performance Plot

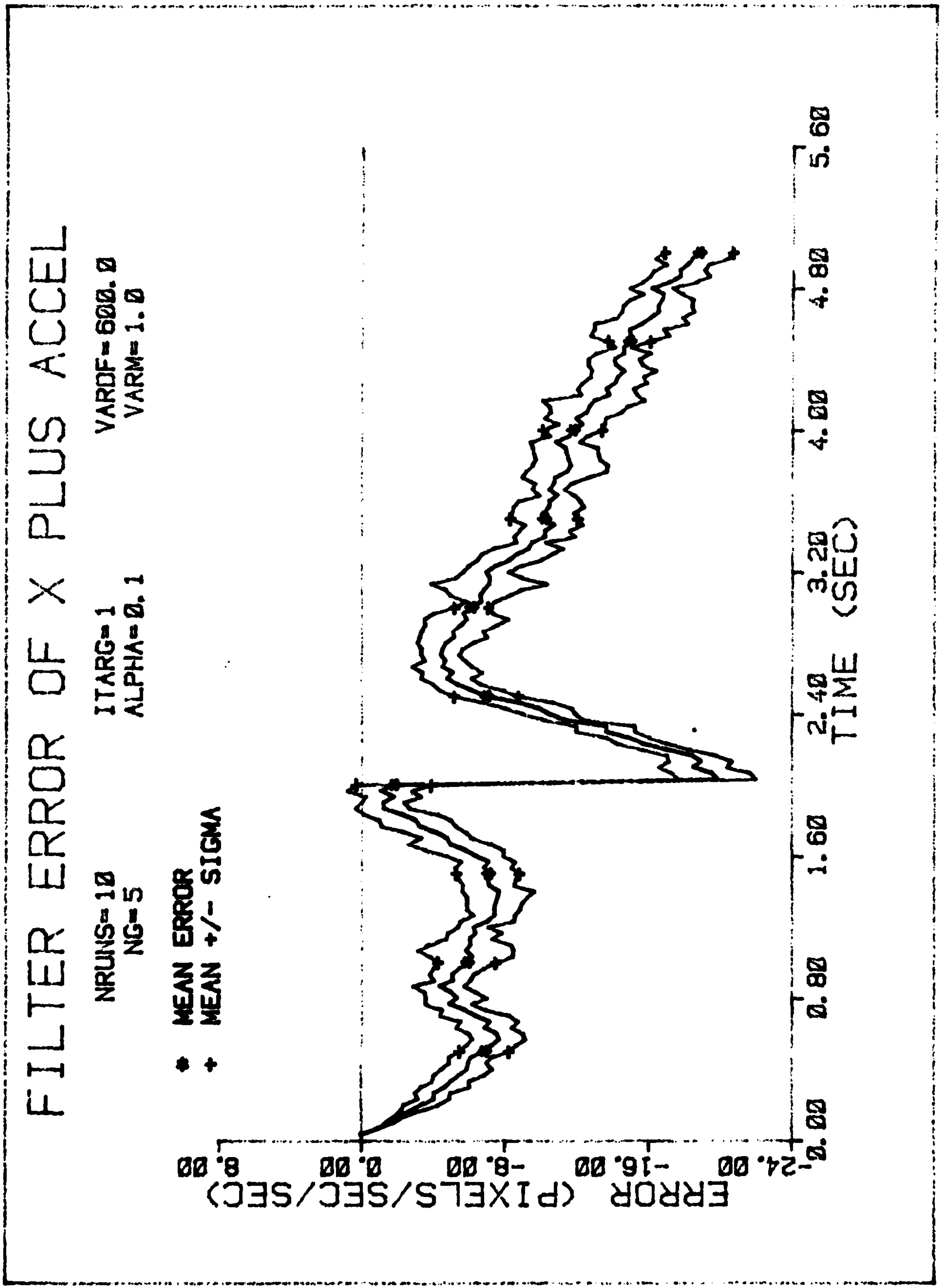


Figure D-98 Case 10 GM Performance Plot

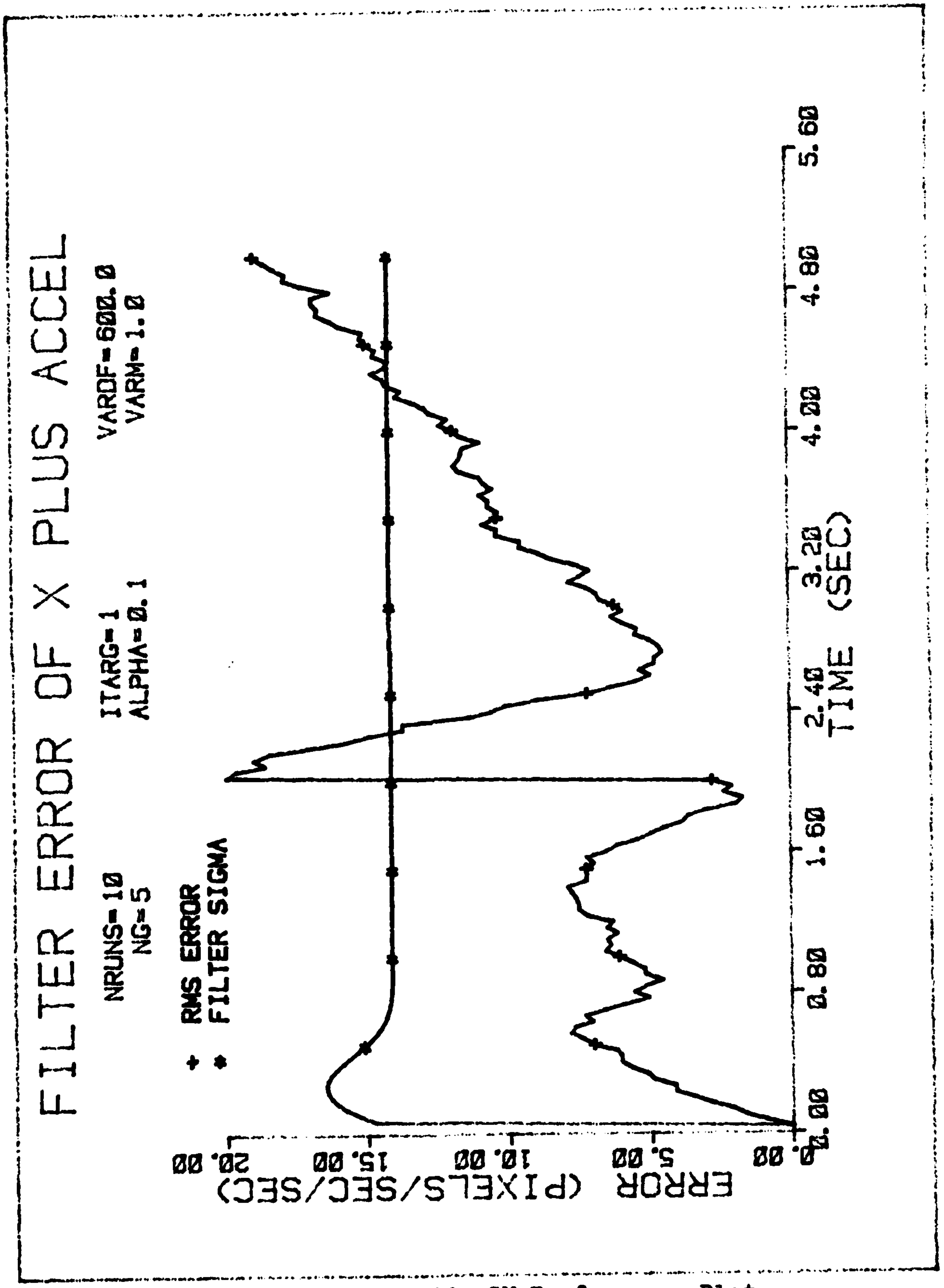


Figure D-99 Case 10 GM Performance Plot

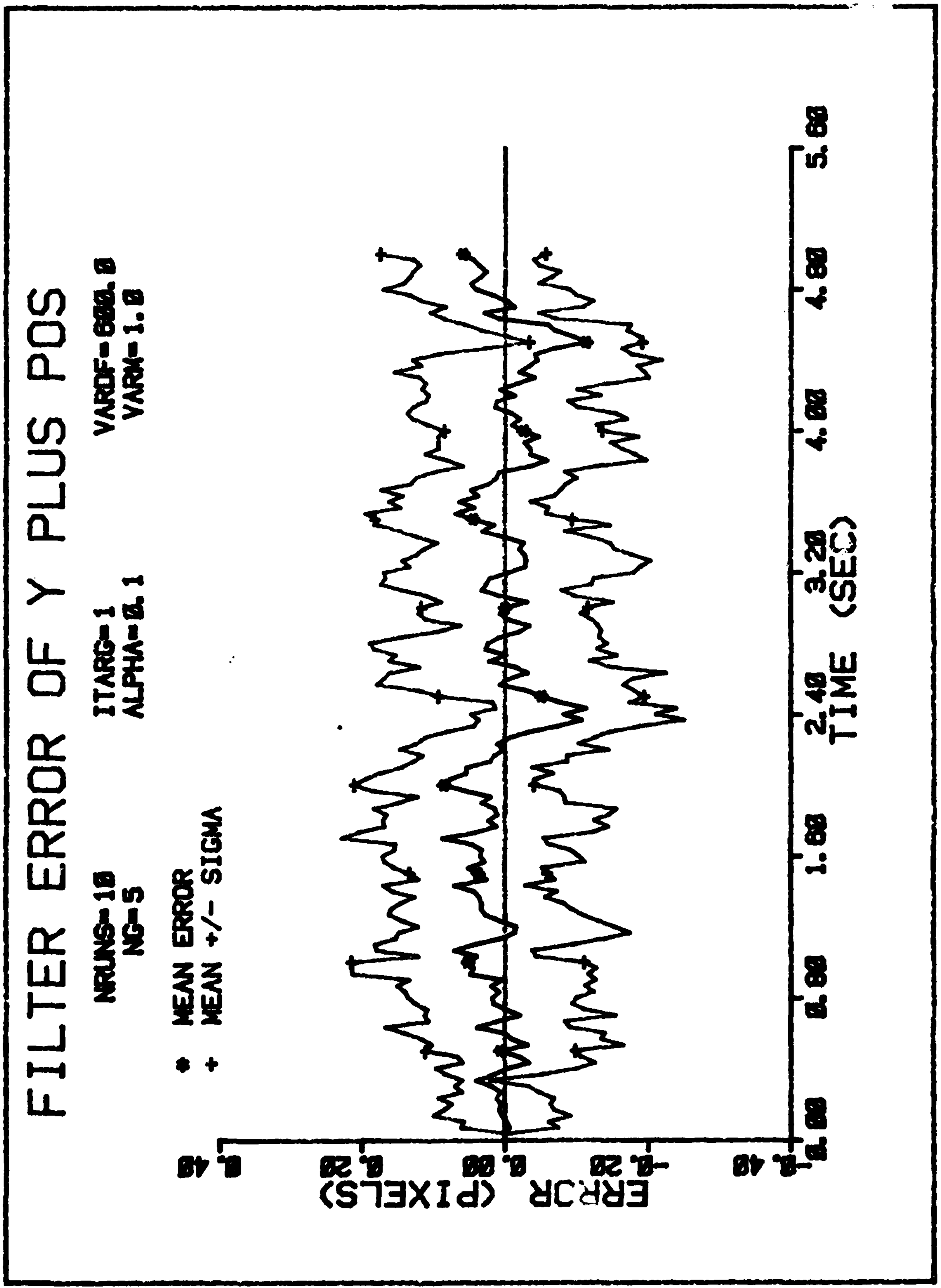


Figure D-100 Case 10 GM Performance Plot

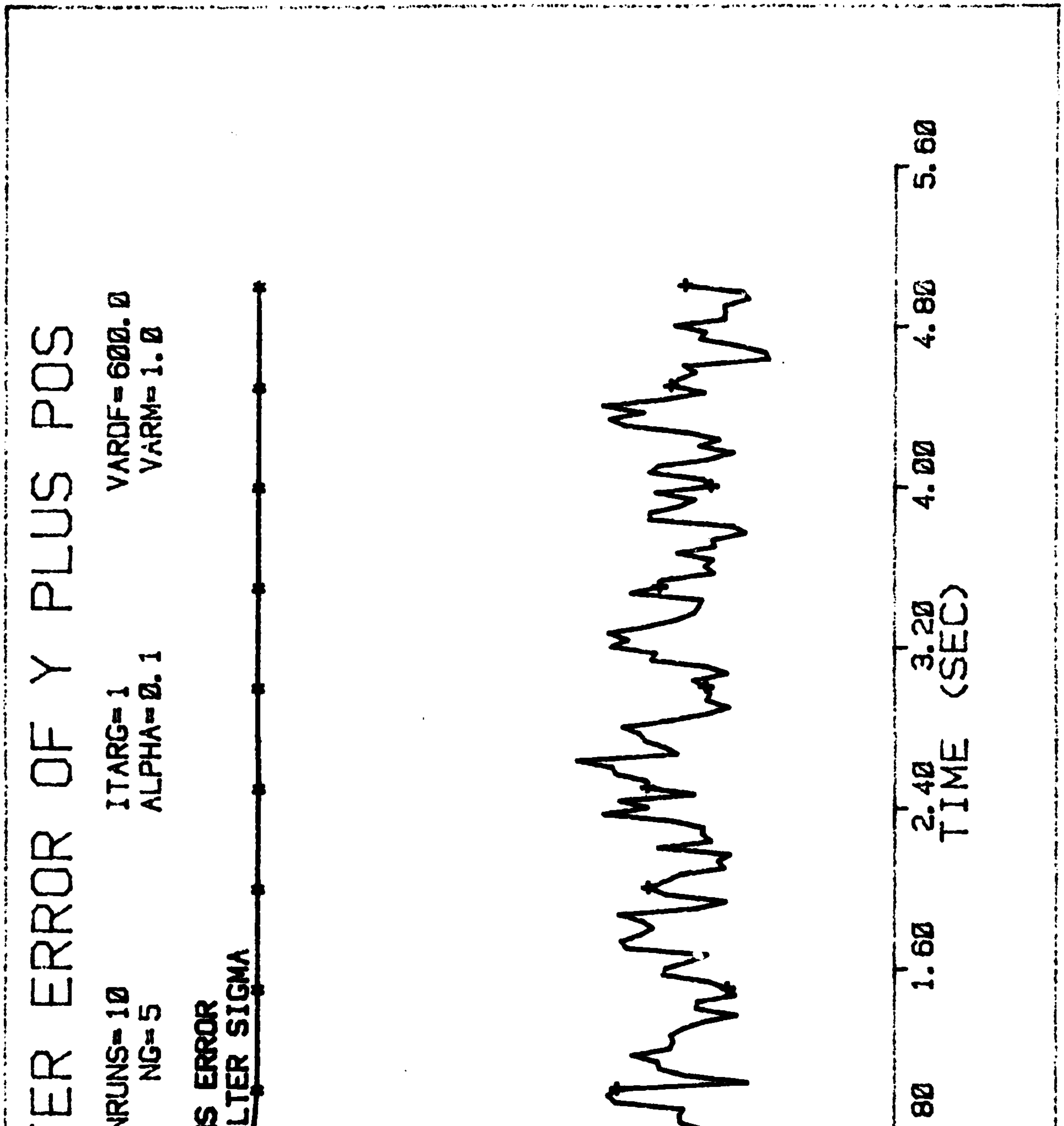


Figure D-101 Case 10 GM Performance Plot

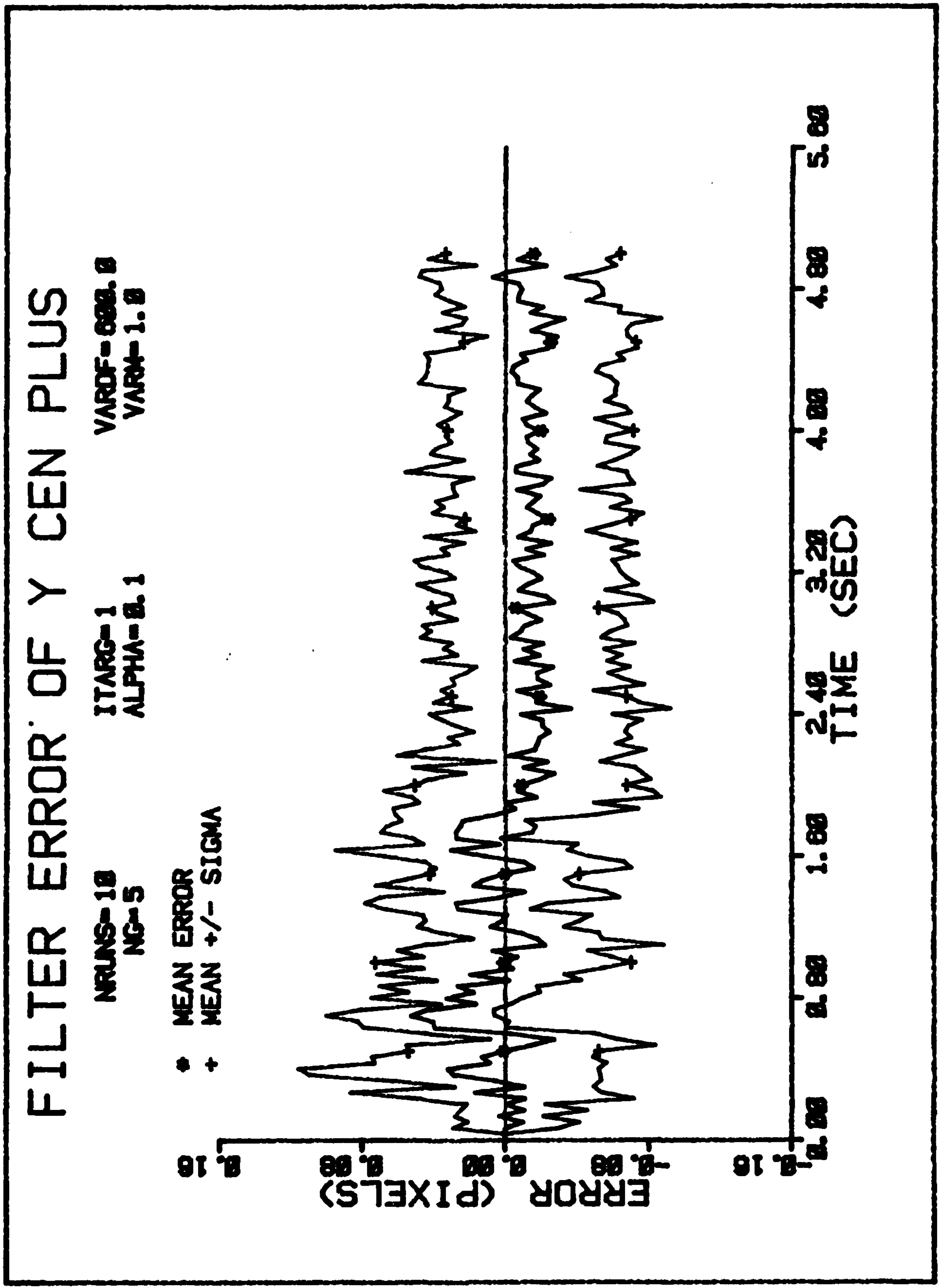


Figure D-102 Case 10 GM Performance Plot

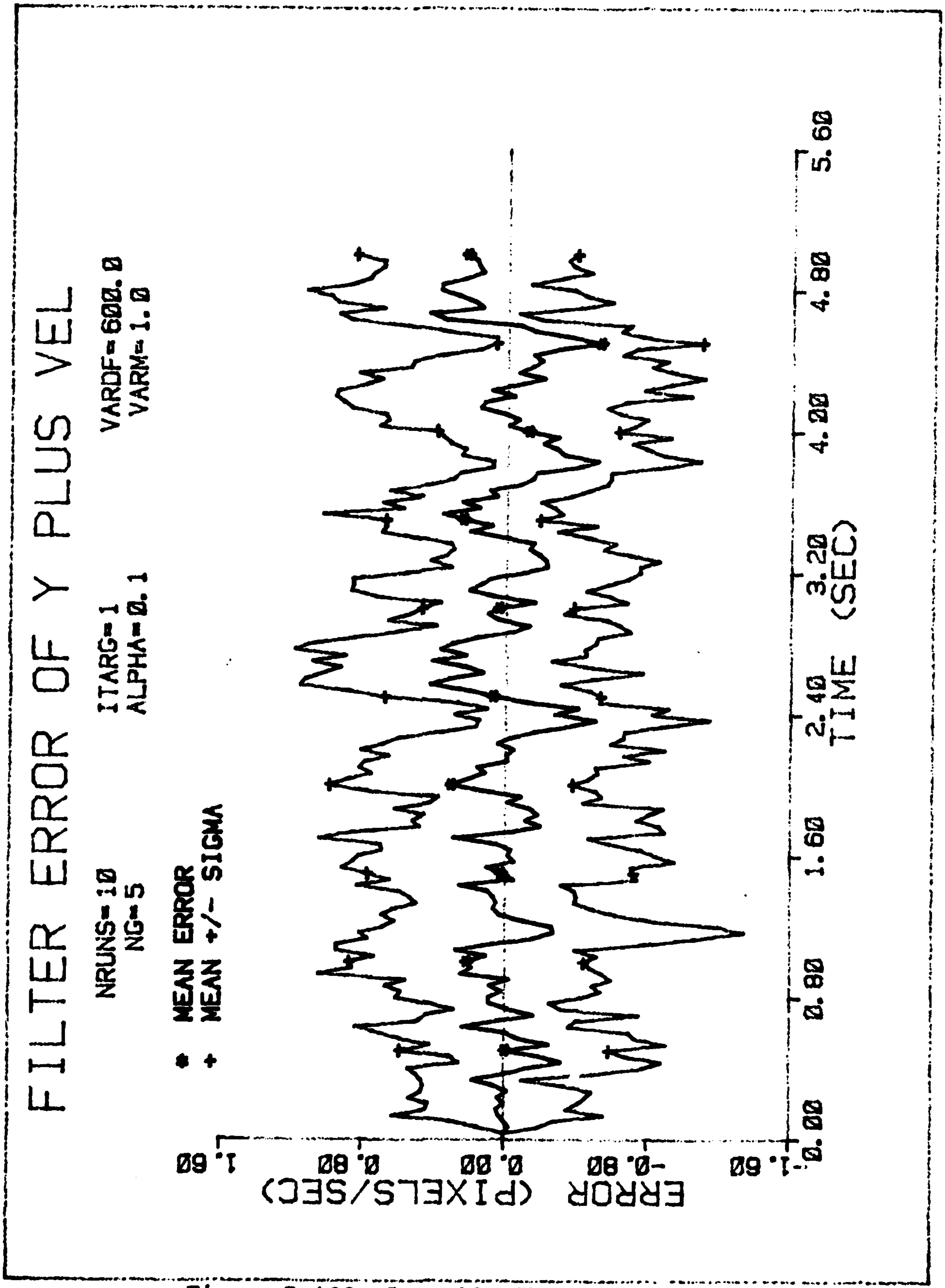


Figure D-103 Case 10 GM Performance Plot

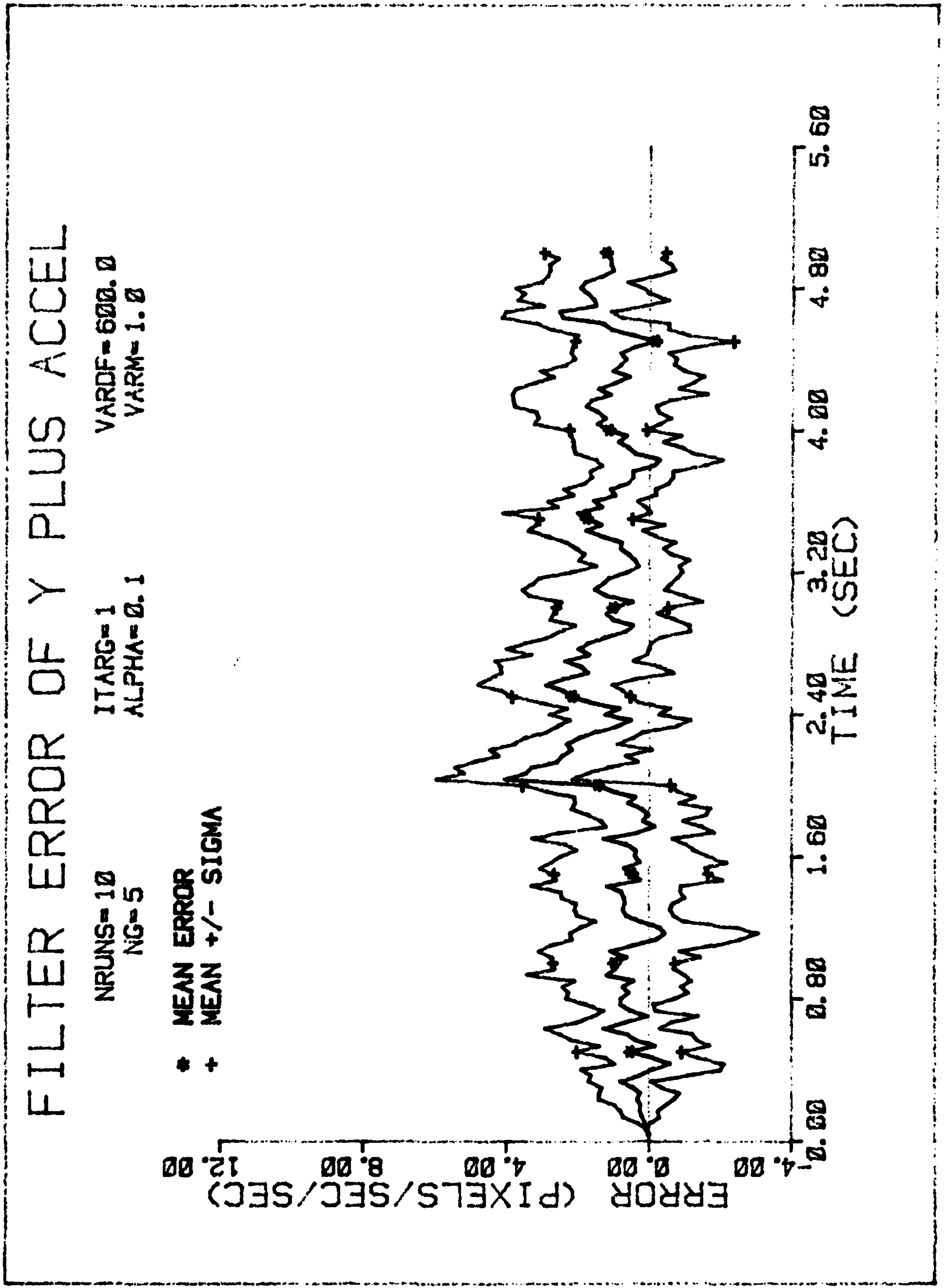


Figure D-104 Case 10 GM Performance Plot

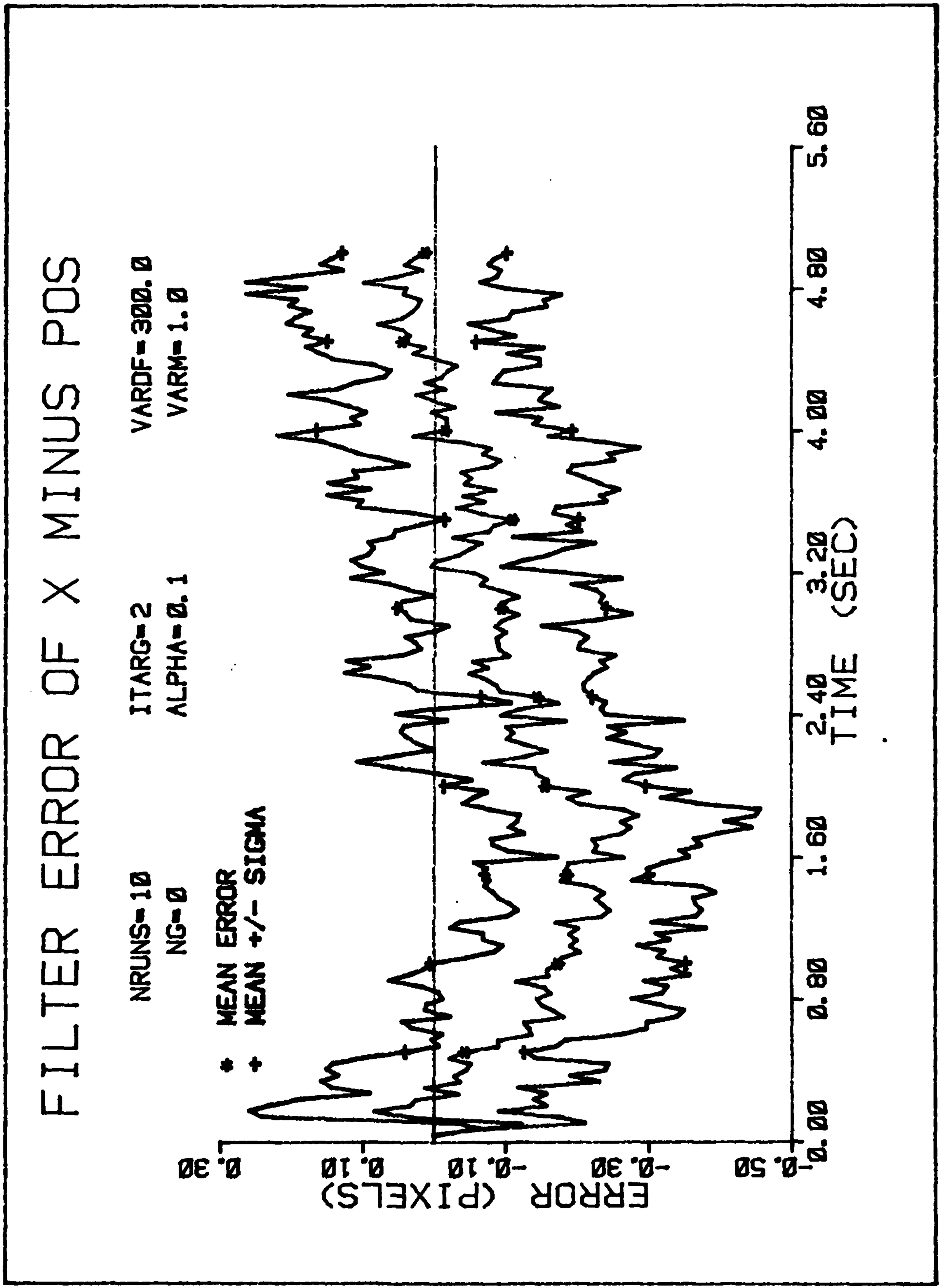


Figure D-105 Case 11 GM Performance Plot

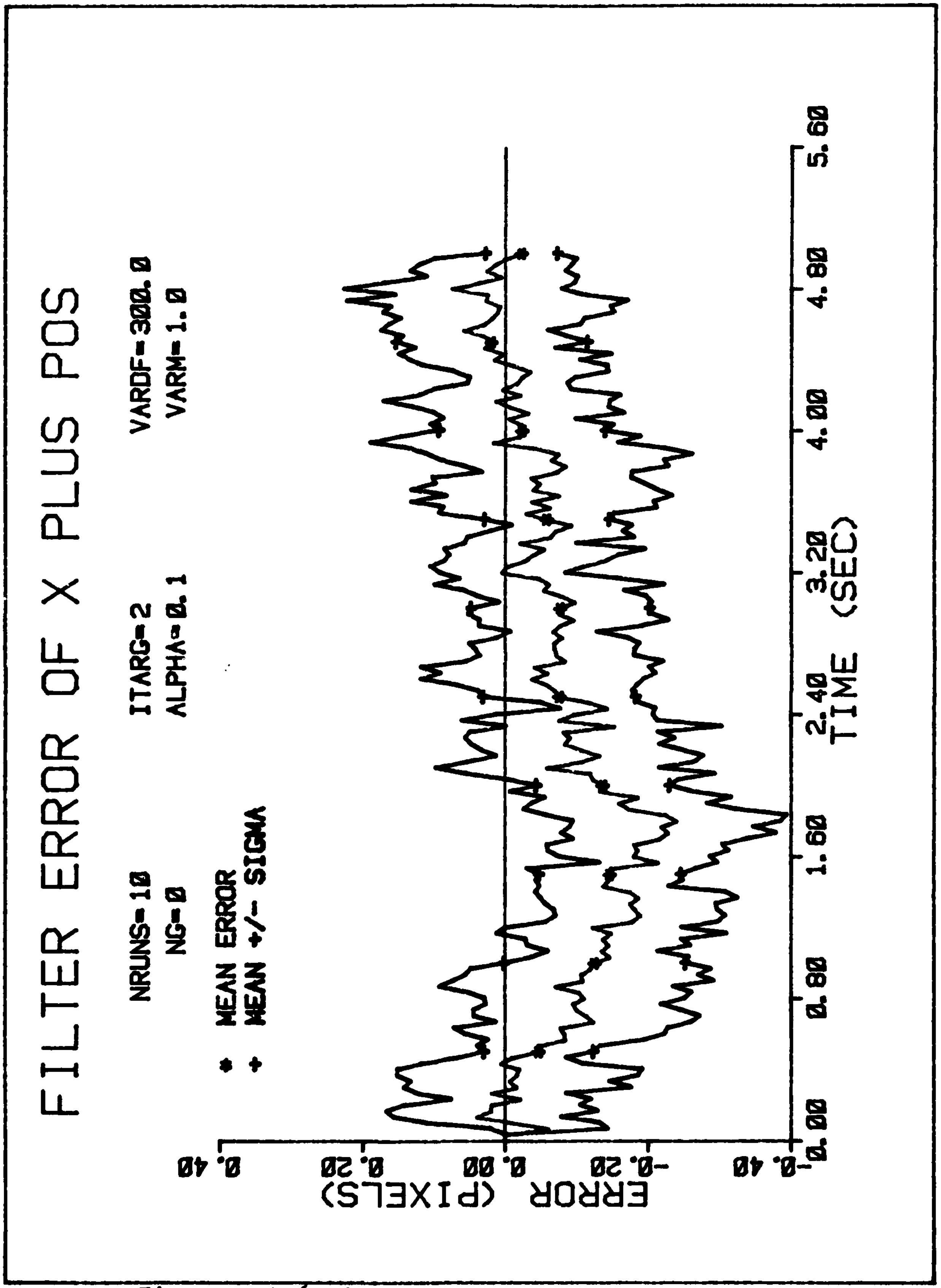


Figure D-106 Case 11 GM Performance Plot

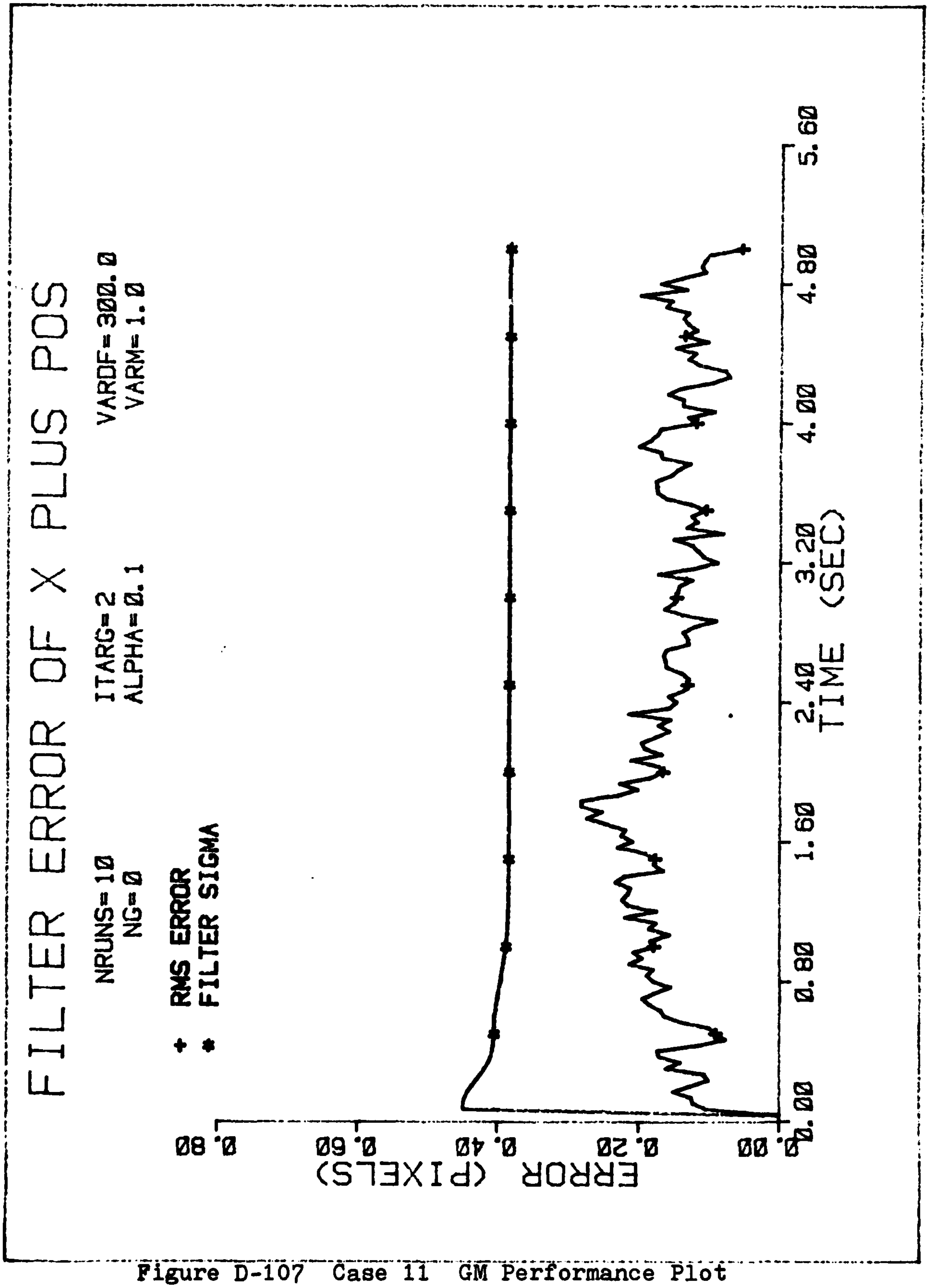


Figure D-107 Case 11 GM Performance Plot

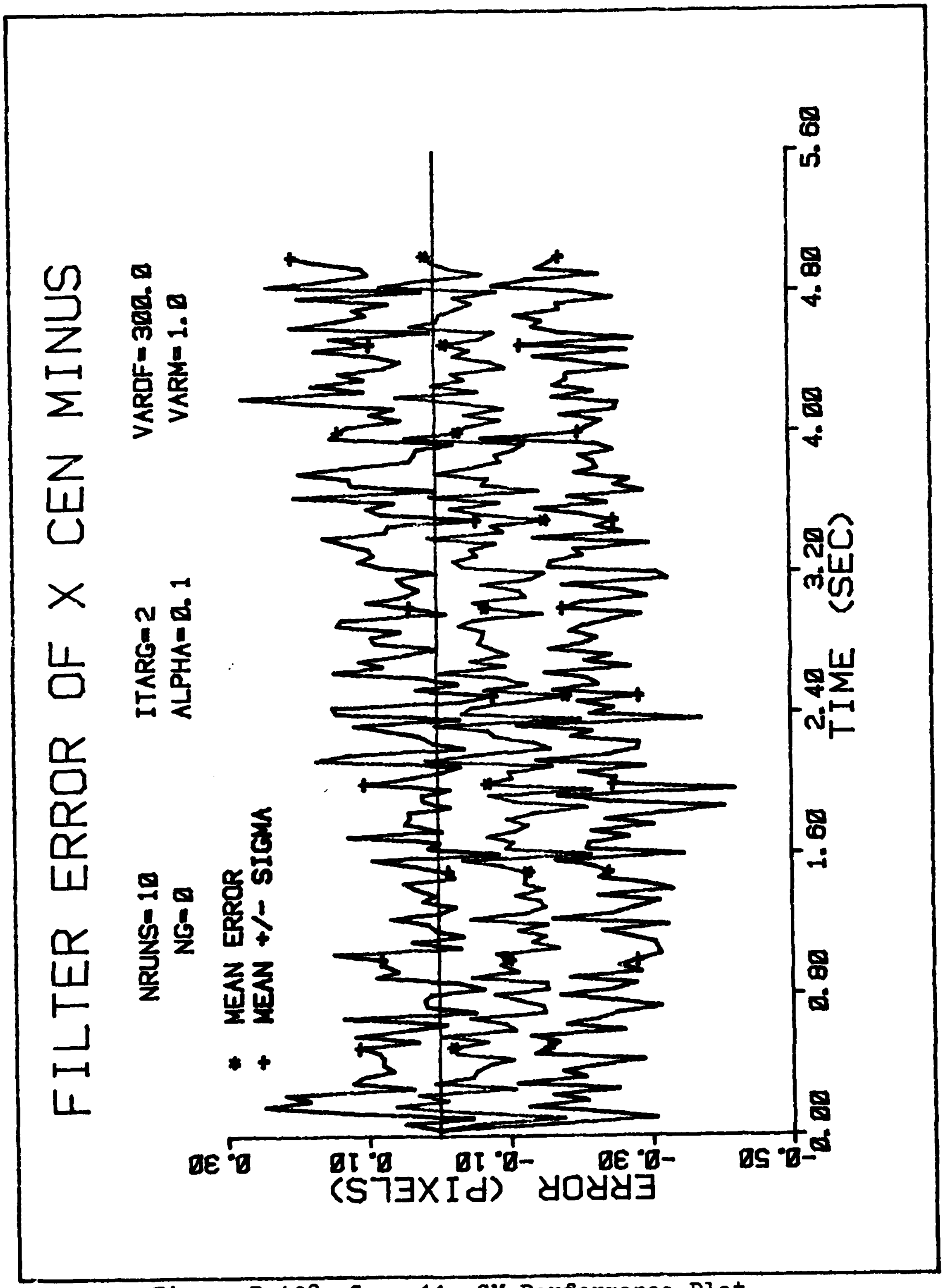


Figure D-108 Case 11 GM Performance Plot

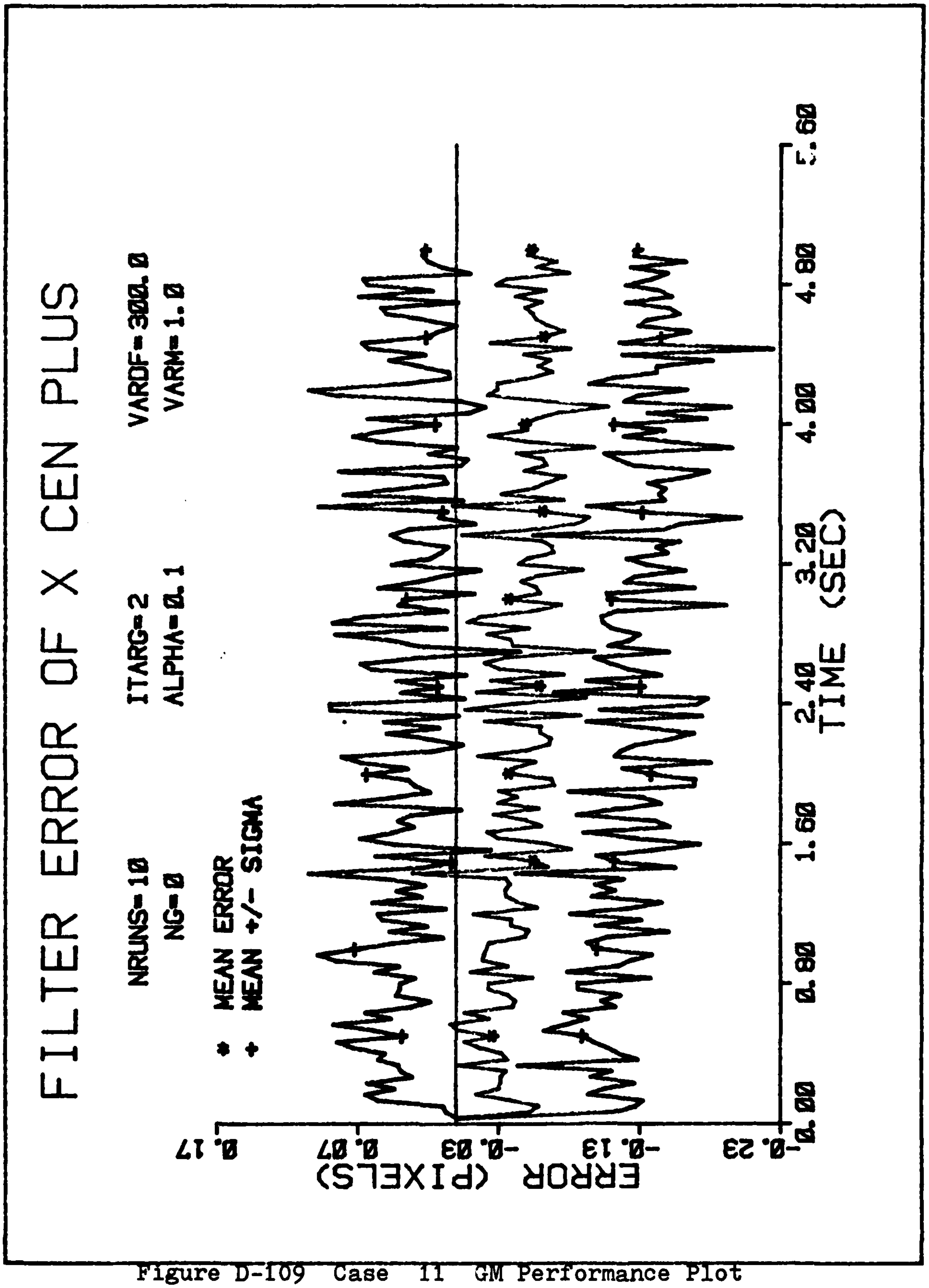


Figure D-109 Case 11 GM Performance Plot

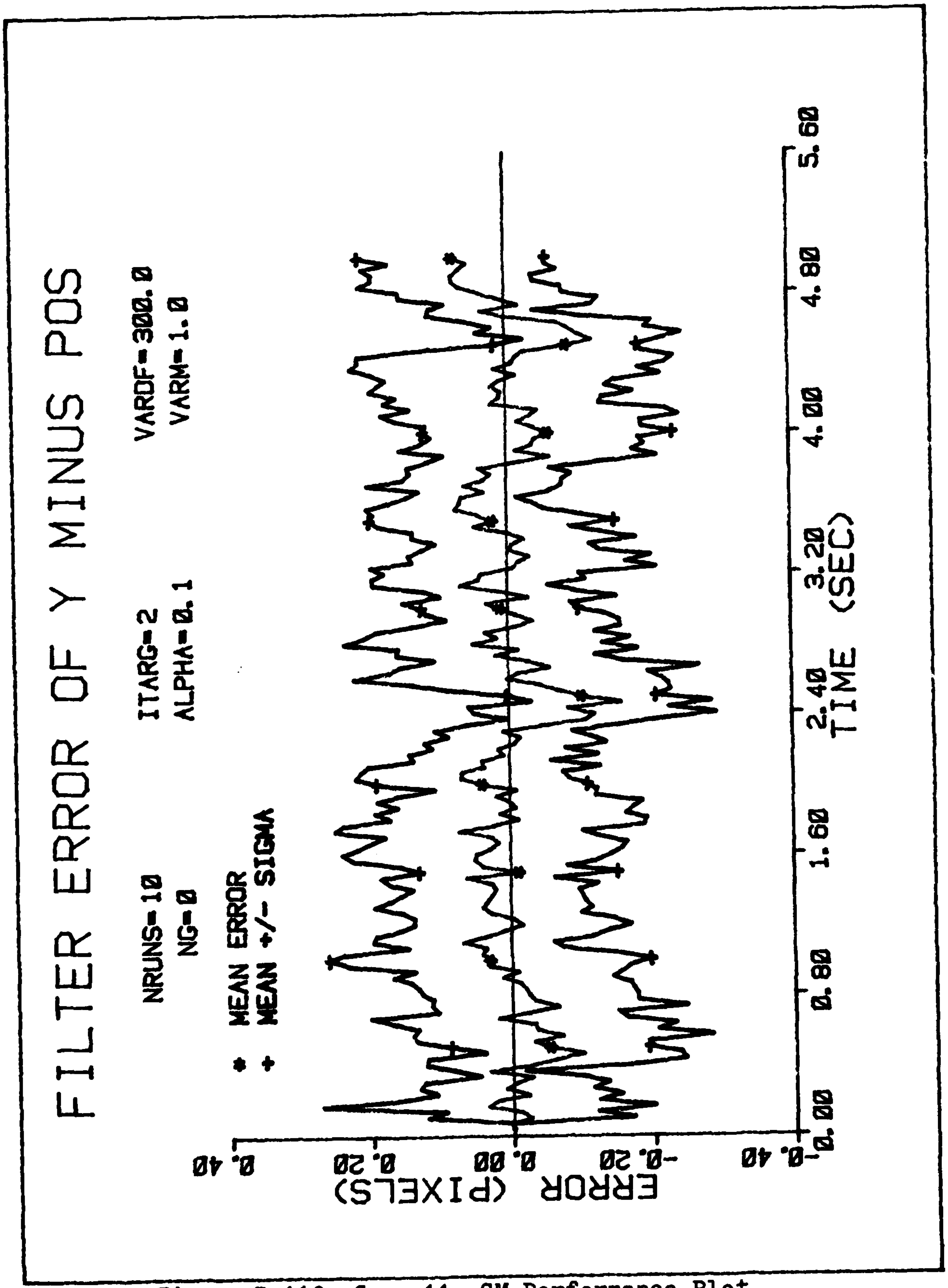


Figure D-110 Case 11 GM Performance Plot

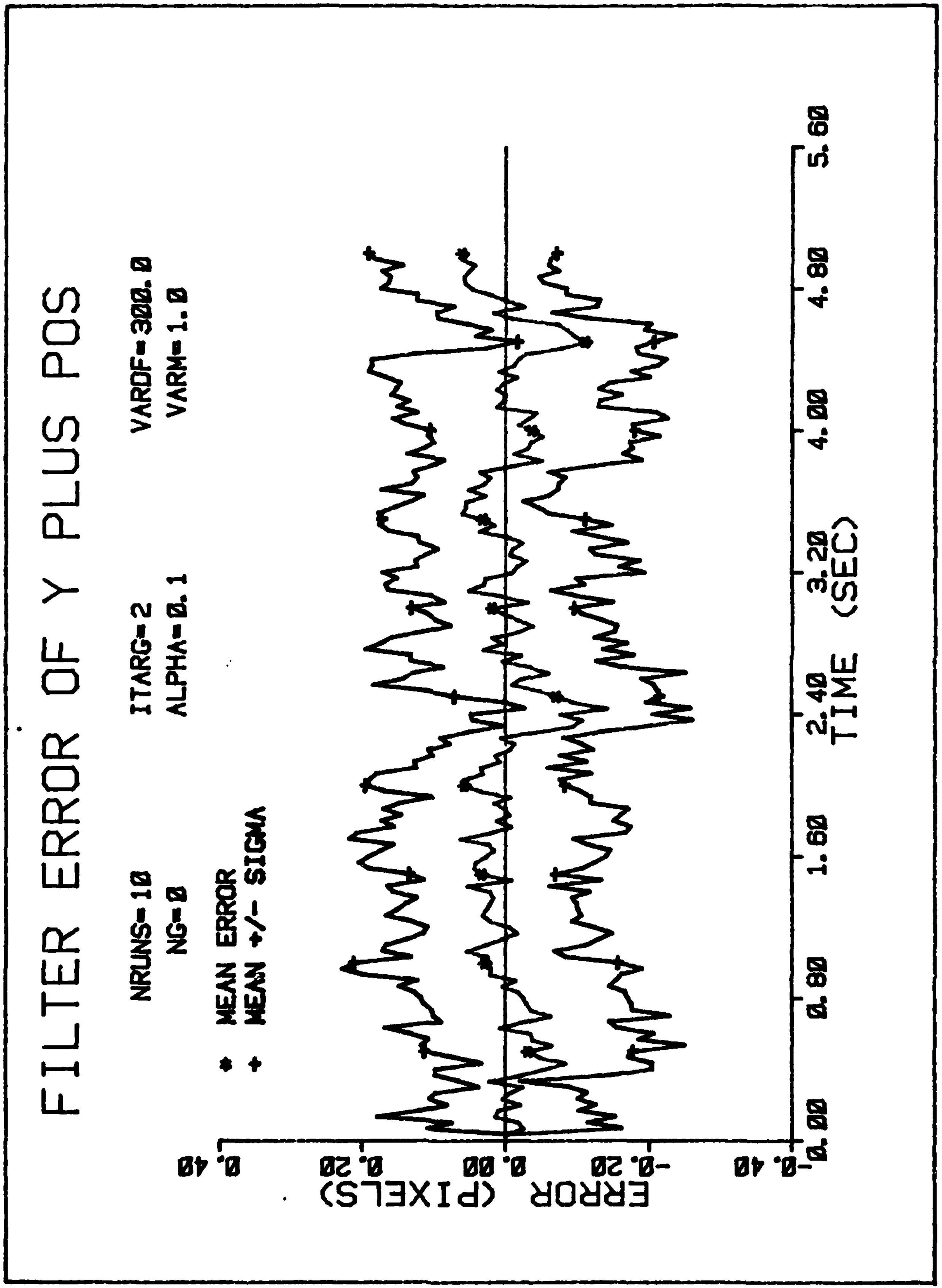


Figure D-111 Case 11 GM Performance Plot

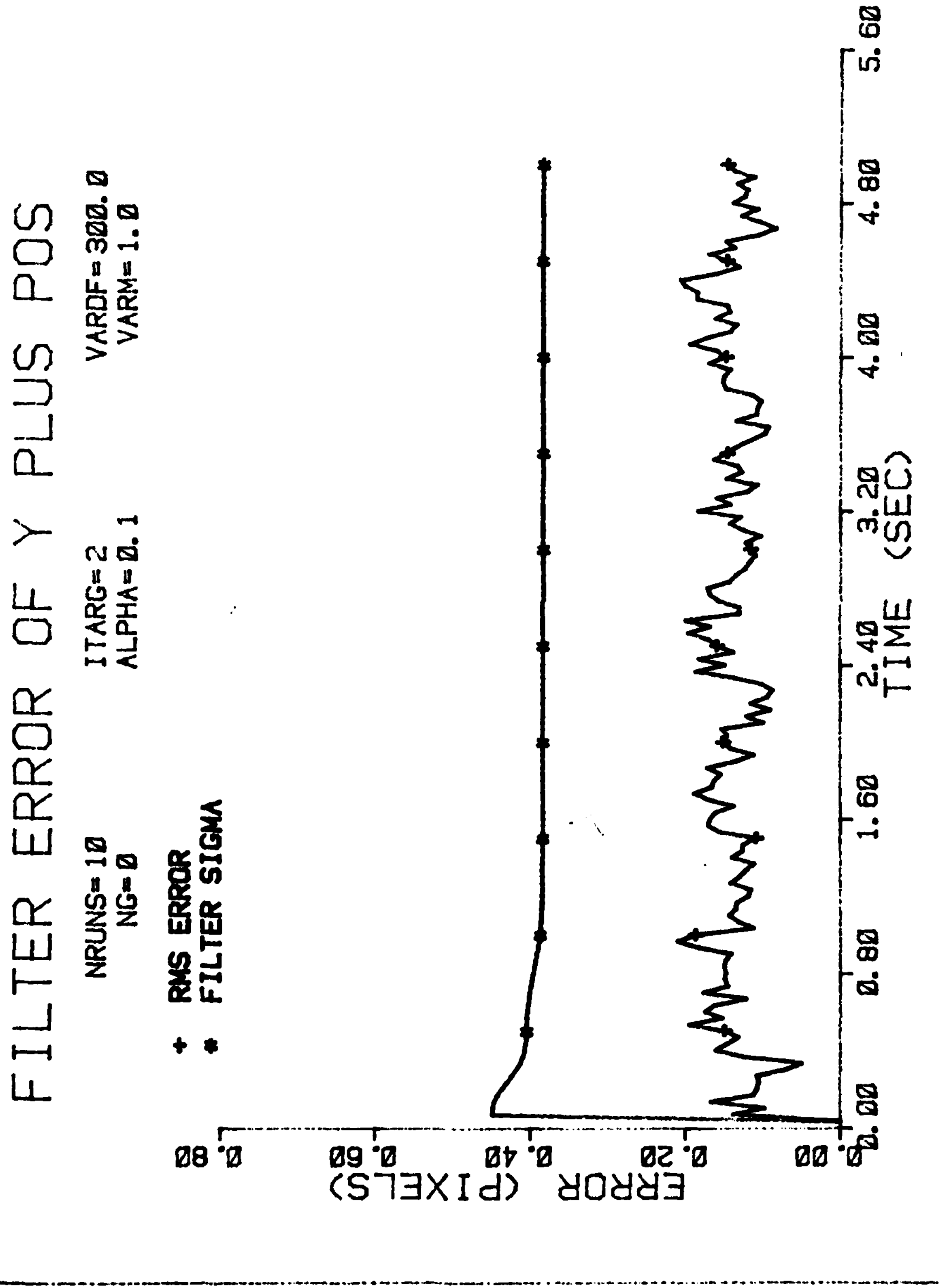


Figure D-112 Case 11 GM Performance Plot

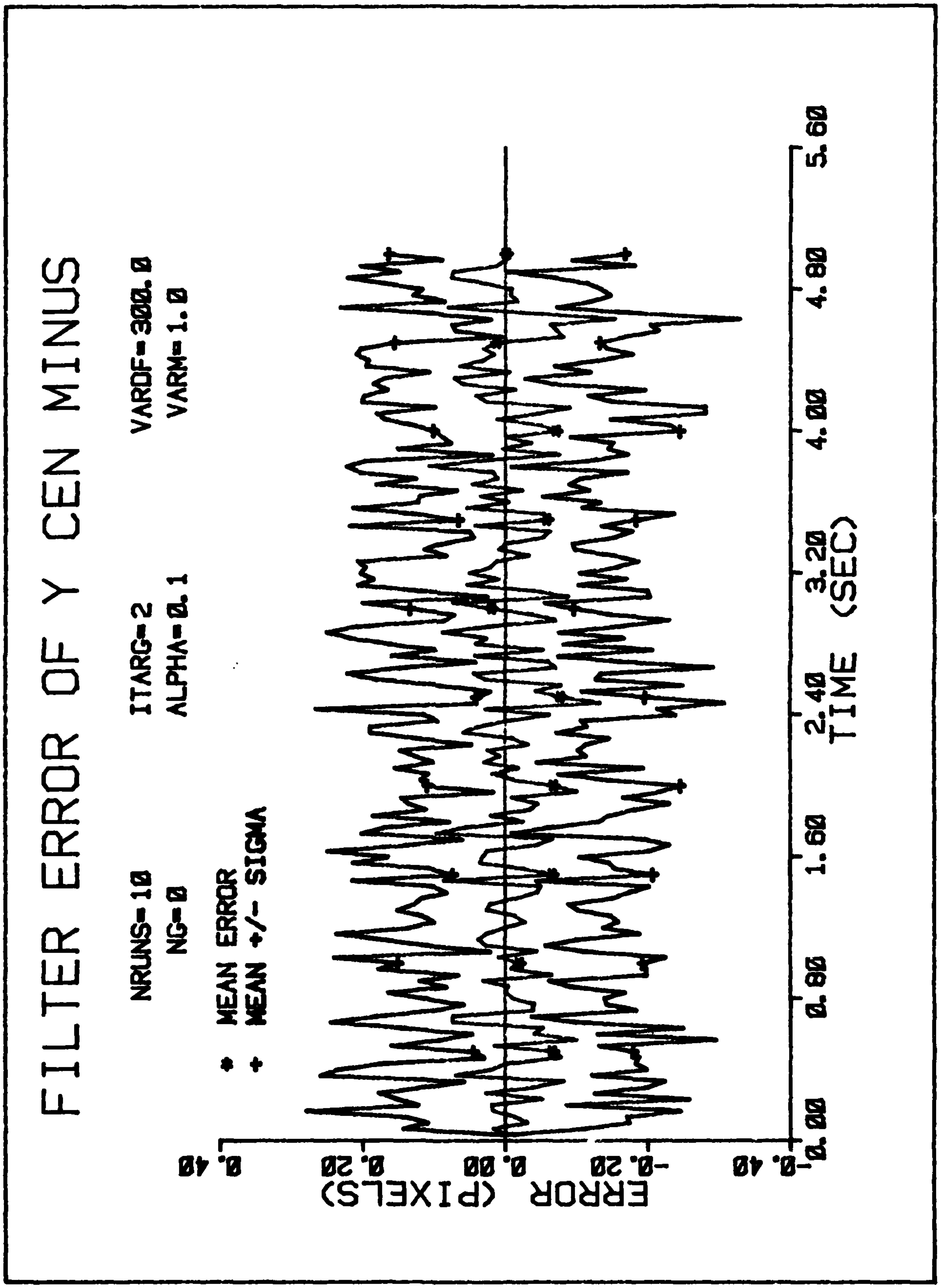


Figure D-113 Case 11 GM Performance Plot

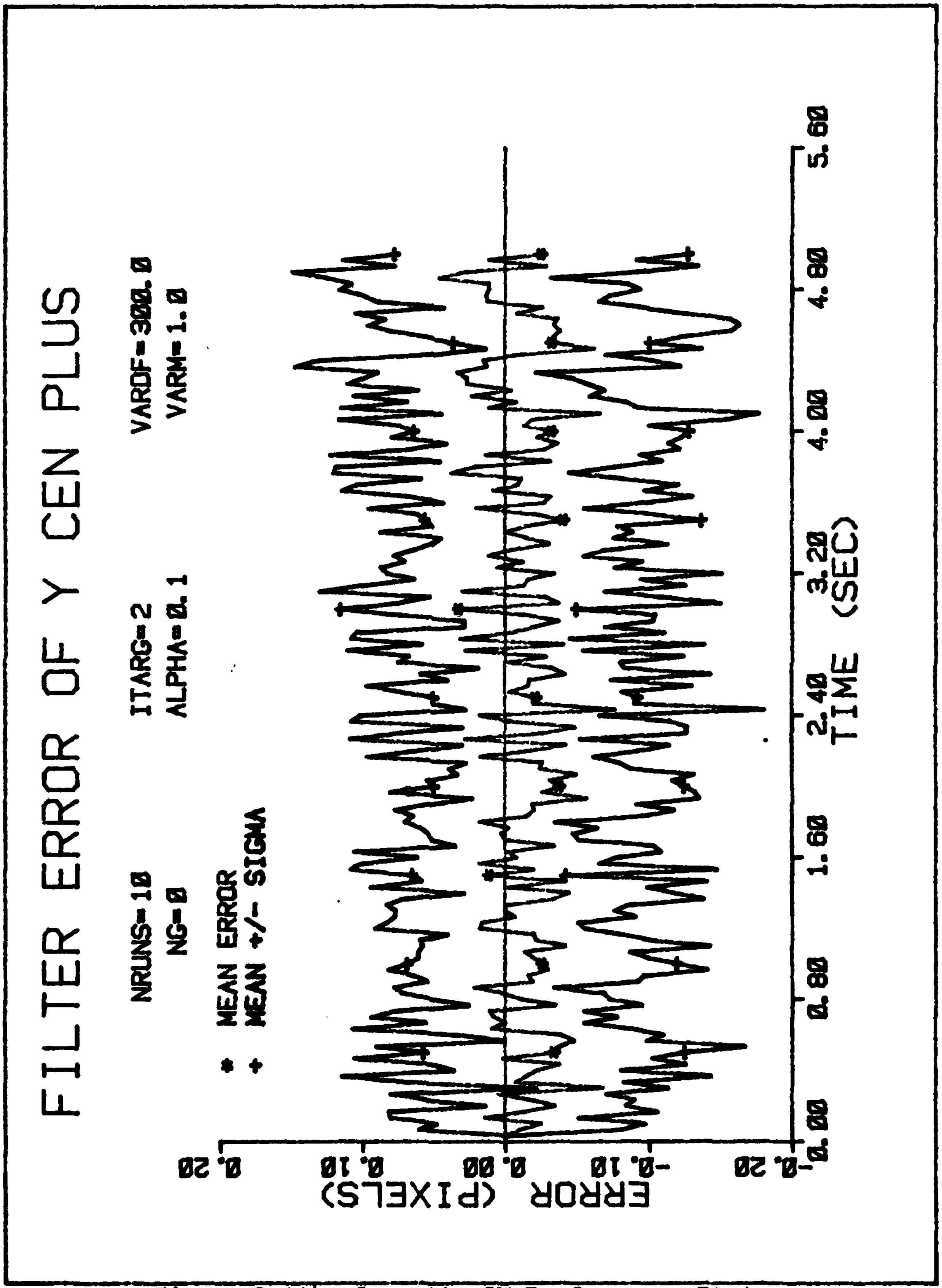


Figure D-114 Case 11 GM Performance Plot

Appendix E

This appendix contains the plotted output for the cases studied that involve the constant turn-rate filter. Each plot is labled with its corresponding case number (12-24). The values of the parameters used in each of these cases are listed in Table E.1.

Table E.1a Parameters of Cases with CTR Filter

	Case												
Param.	12	13	14	15	16	17	18	19	20	21	22	23	24
Model	CTR	CTR	CTR	CTR	CTR	CTR	CTR	CTR	CTR	CTR	CTR	CTR	CTR
Traj.	2	1	1	3	3	3	3	4	1	1	1	3	3
REVRT	0	0	0	0.5	0.5	0.5	0	0.5	1.0	0.5	1.0	0	0.5
NG	0	0	0	2	2	5	5	5	0	0	0	2	2
ITARG	2	0	2	1	1	1	1	1	1	1	1	0	1
VARDF	160	300	300	300	600	1000	5000	600	300	300	300	300	300
SIGAT	.141	.141	.141	.100	.100	.141	.141	.141	.141	.141	.141	.100	.141
VARAF	.200	.200	.200	.100	.100	.200	.200	.200	.200	.200	.200	.100	.200
VARM	1	1	1	1	1	1	1	1	1	1	1	1	1
Filter	2	2	2	2	2	2	2	2	2	2	2	2	2
COV	3	2	2	-	-	-	-	-	-	-	-	2	-
ALPHA	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.2	0.1	0.1
VXO	-700	-1000	-1000	-1000	-1000	-1000	-1000	-1000	-1000	-1000	-1000	-1000	-1000
VYO	-300	0	0	0	0	0	0	0	0	0	0	0	0
VZO	500	0	0	0	0	0	0	0	0	0	0	0	0
RFIL	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table E.1b Parameters Held Constant

Parameter	Value
IMAX	20
NF	0
NZ	0
NFRMS	150
NRUNS	10
(XO, YO, ZO)	(5000, 500, 20000)
(AXO, AYO, AZO)	(0, 0, 0)
SIGVO	(4.0, 1.5, 1.5)
SIGPVO	(1.0, 1.0, 1.0)
DELV	(.64, -.86, -.86)
DELPV	(0.0, 2.5, -2.5)
TDF	1.5
TAF	0.07072

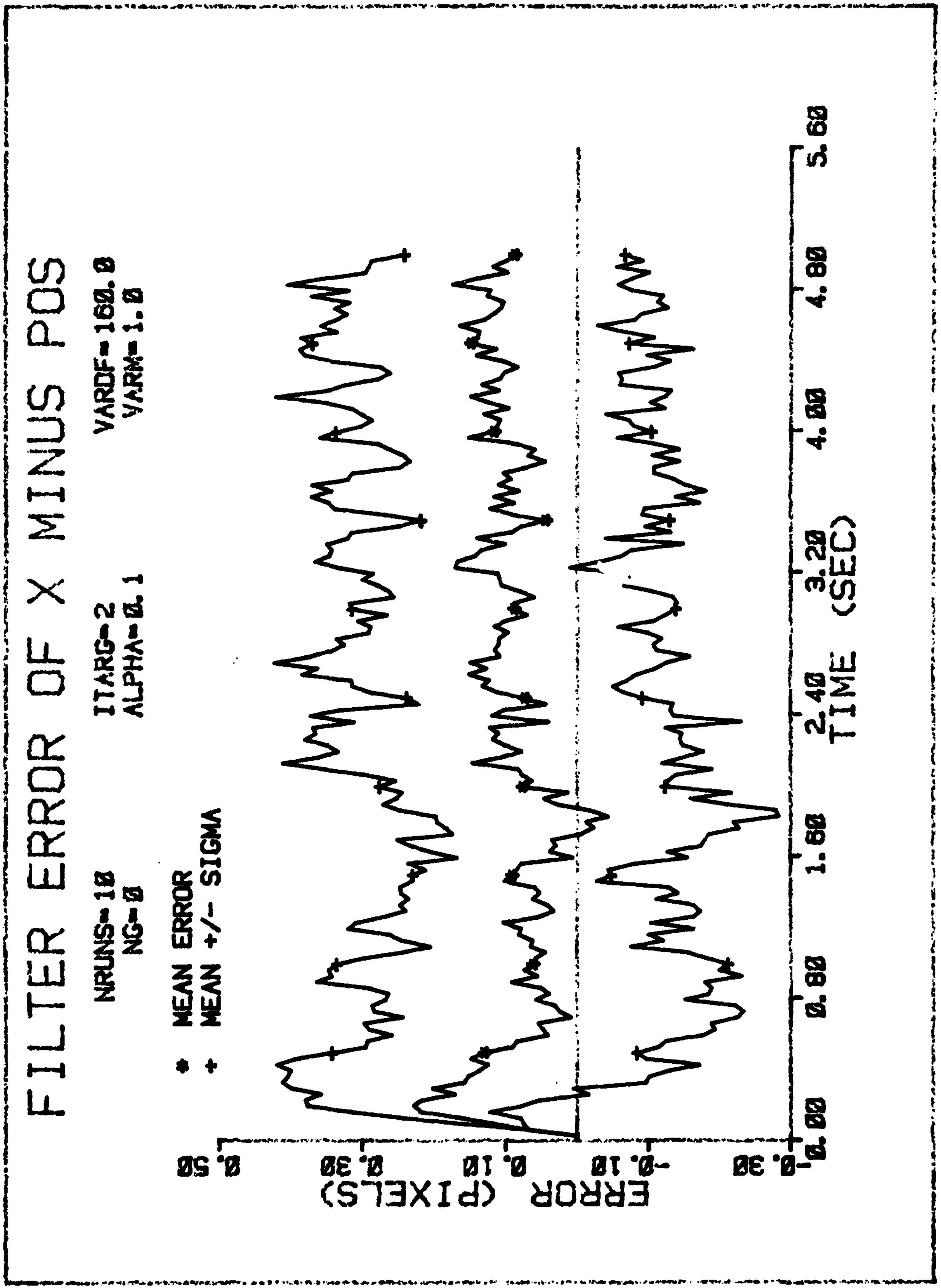


Figure E-1 Case 12 CTR Performance Plot

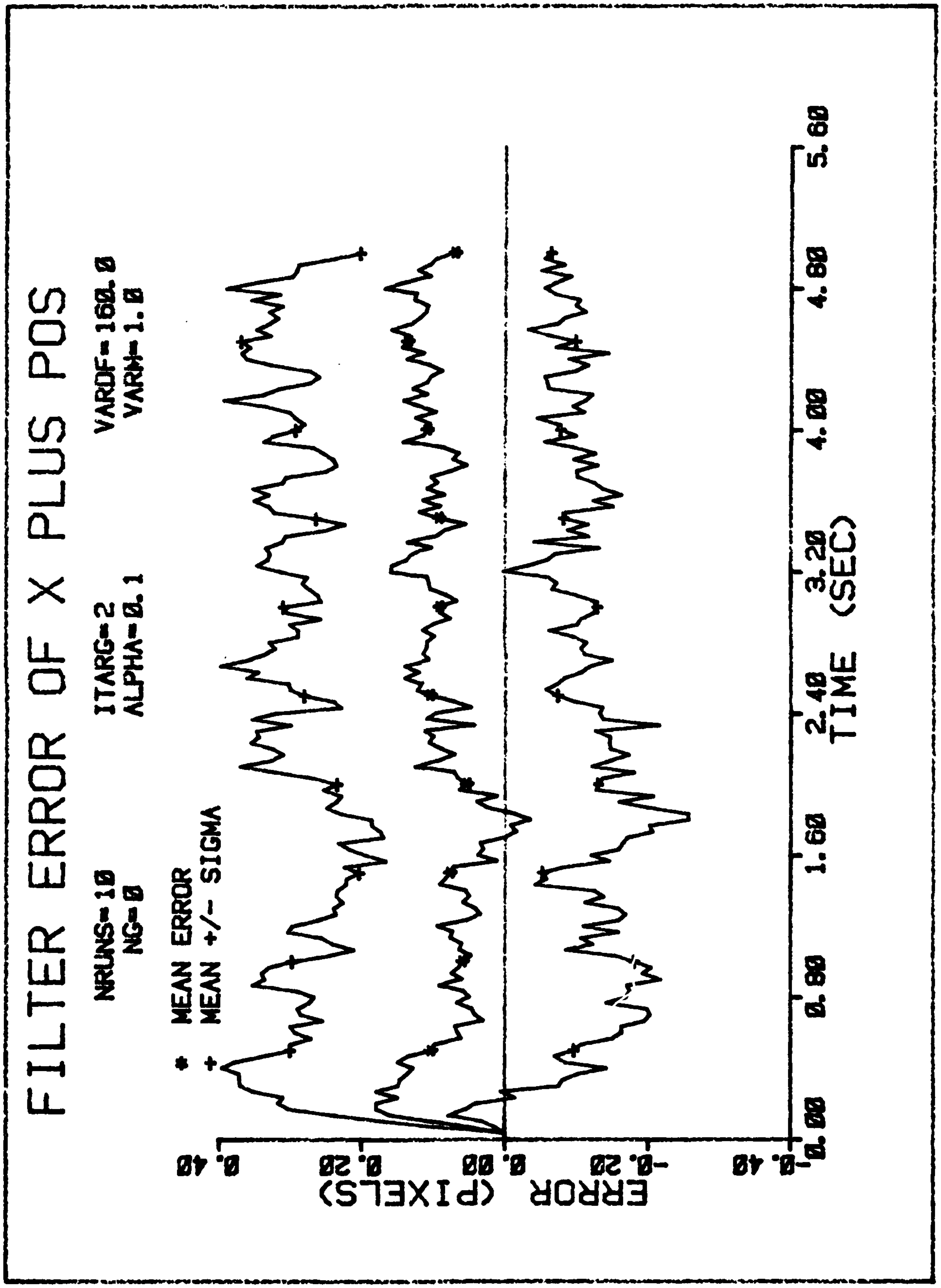


Figure E-2 Case 12 CTR Performance Plot

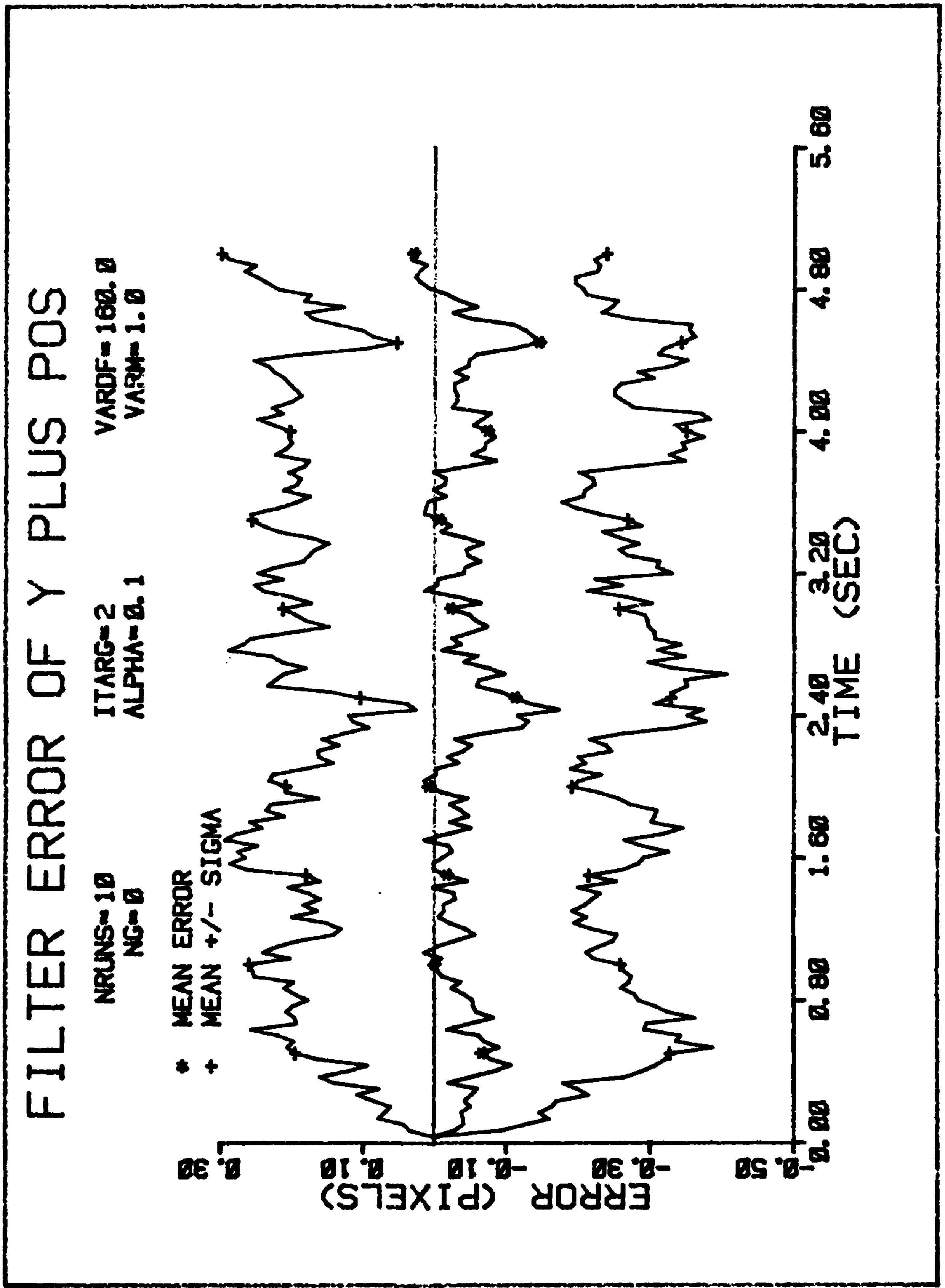
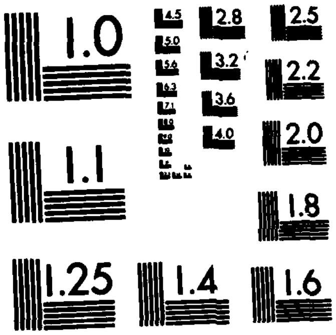


Figure E-10 Case 12 CTR Performance Plot

AD-A124 781 ENHANCED IMAGE TRACKING: ANALYSIS OF TWO ACCELERATION 4/6
MODELS IN TRACKING..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. M R KOZEMCHAK
UNCLASSIFIED DEC 82 AFIT/GEO/EE/82D-4 F/G 17/7 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

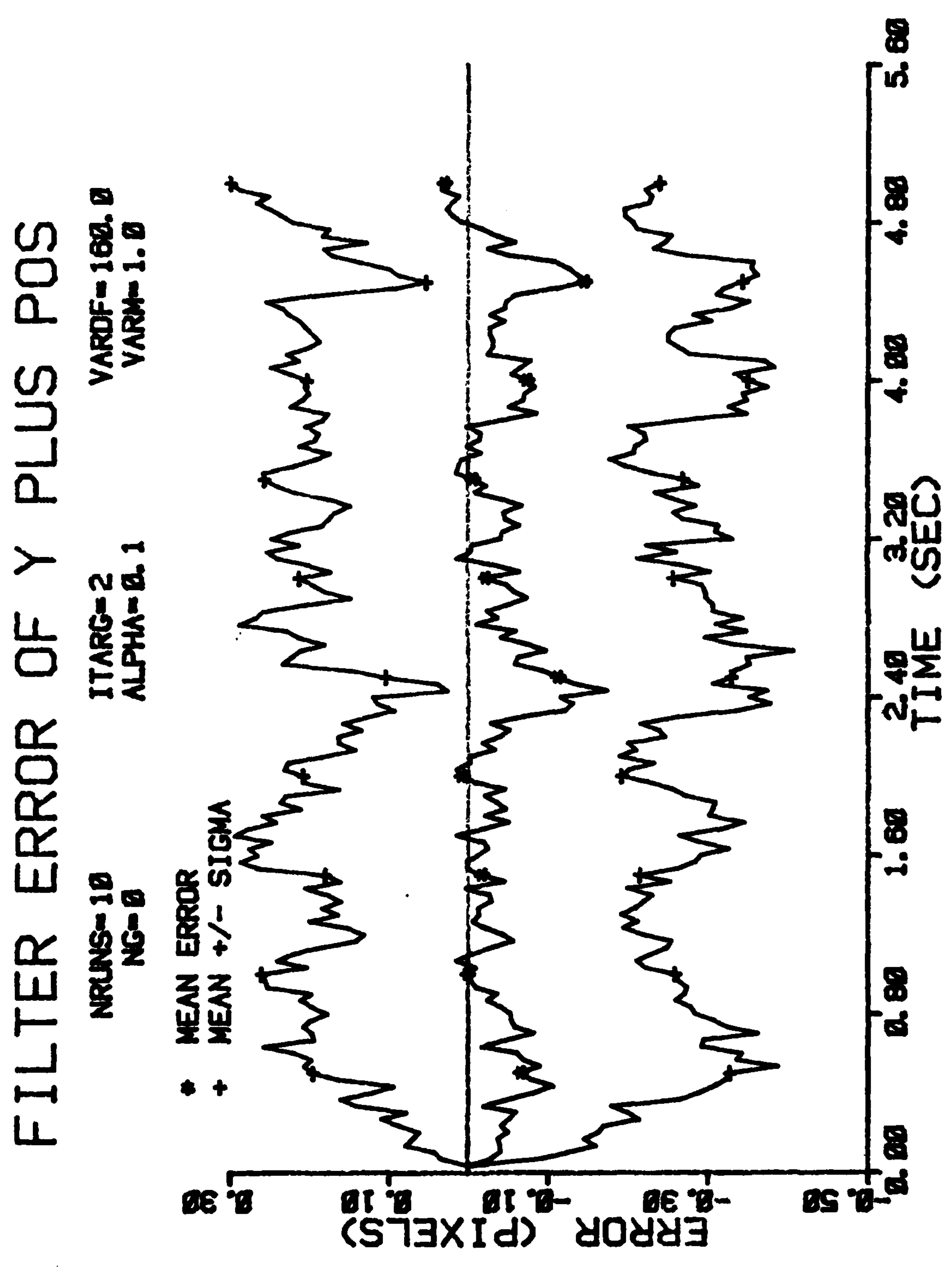


Figure E-10 Case 12 CTR Performance Plot

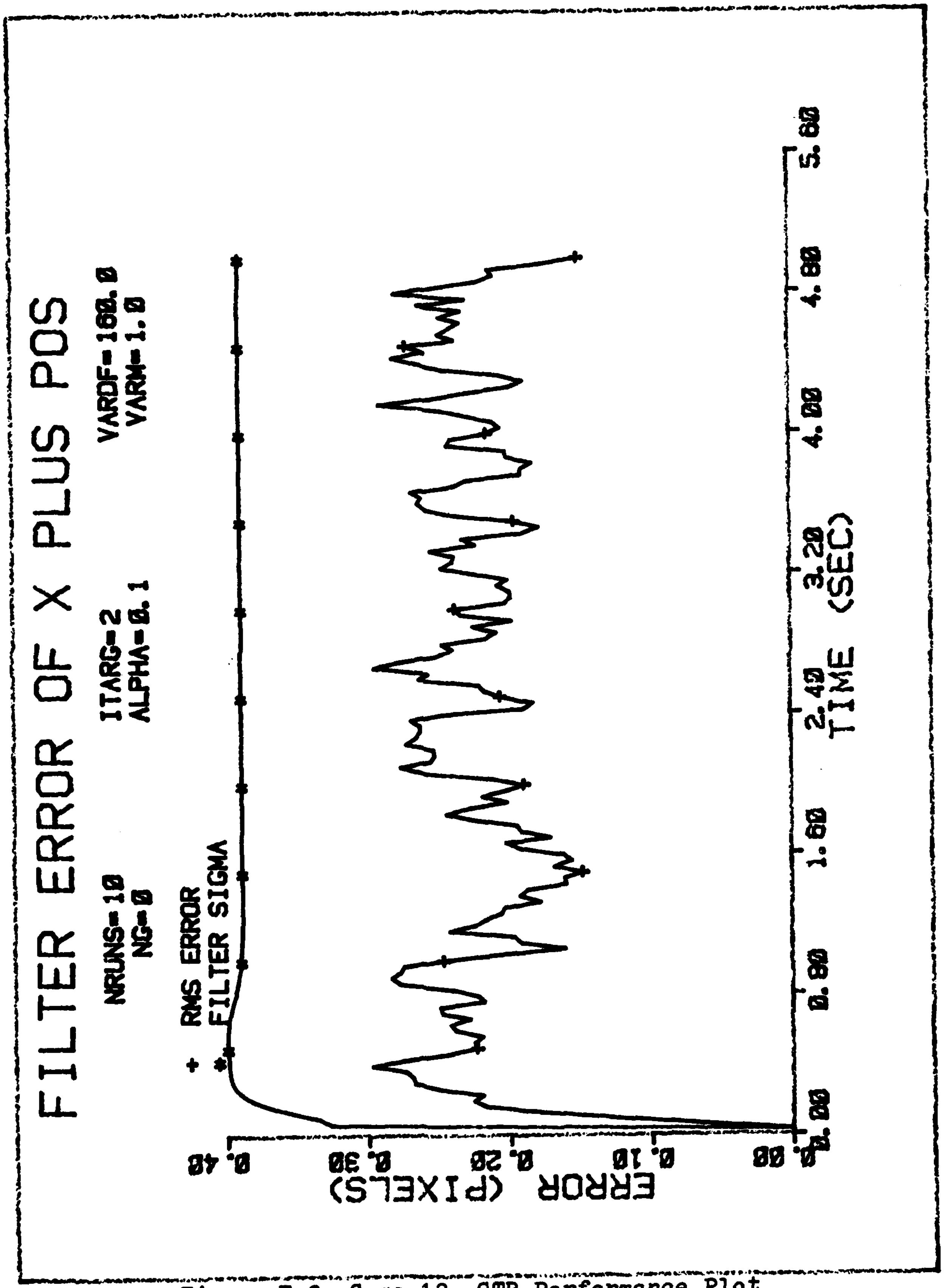


Figure E-3 Case 12 CTR Performance Plot

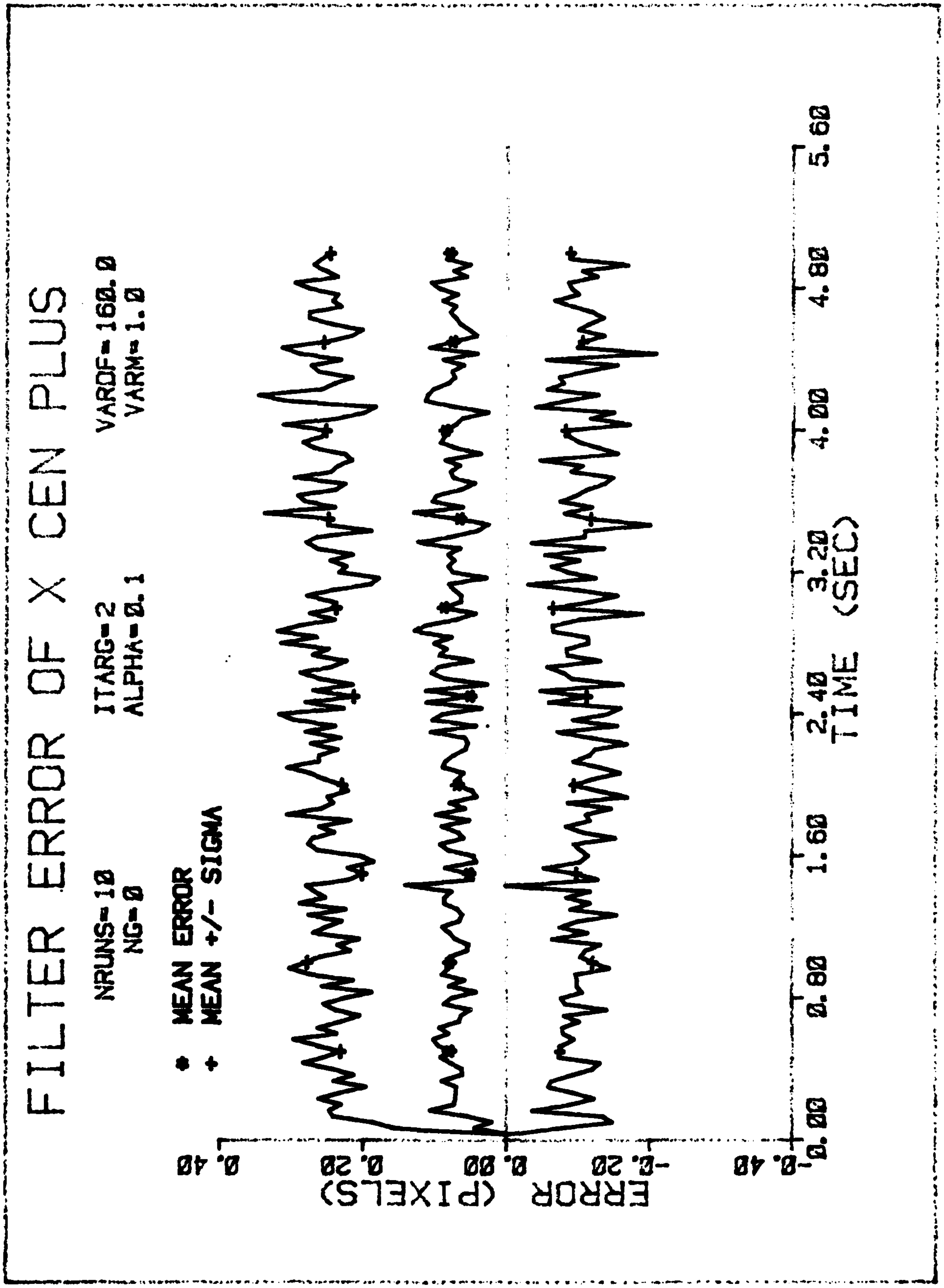


Figure E-4 Case 12 CTR Performance Plot

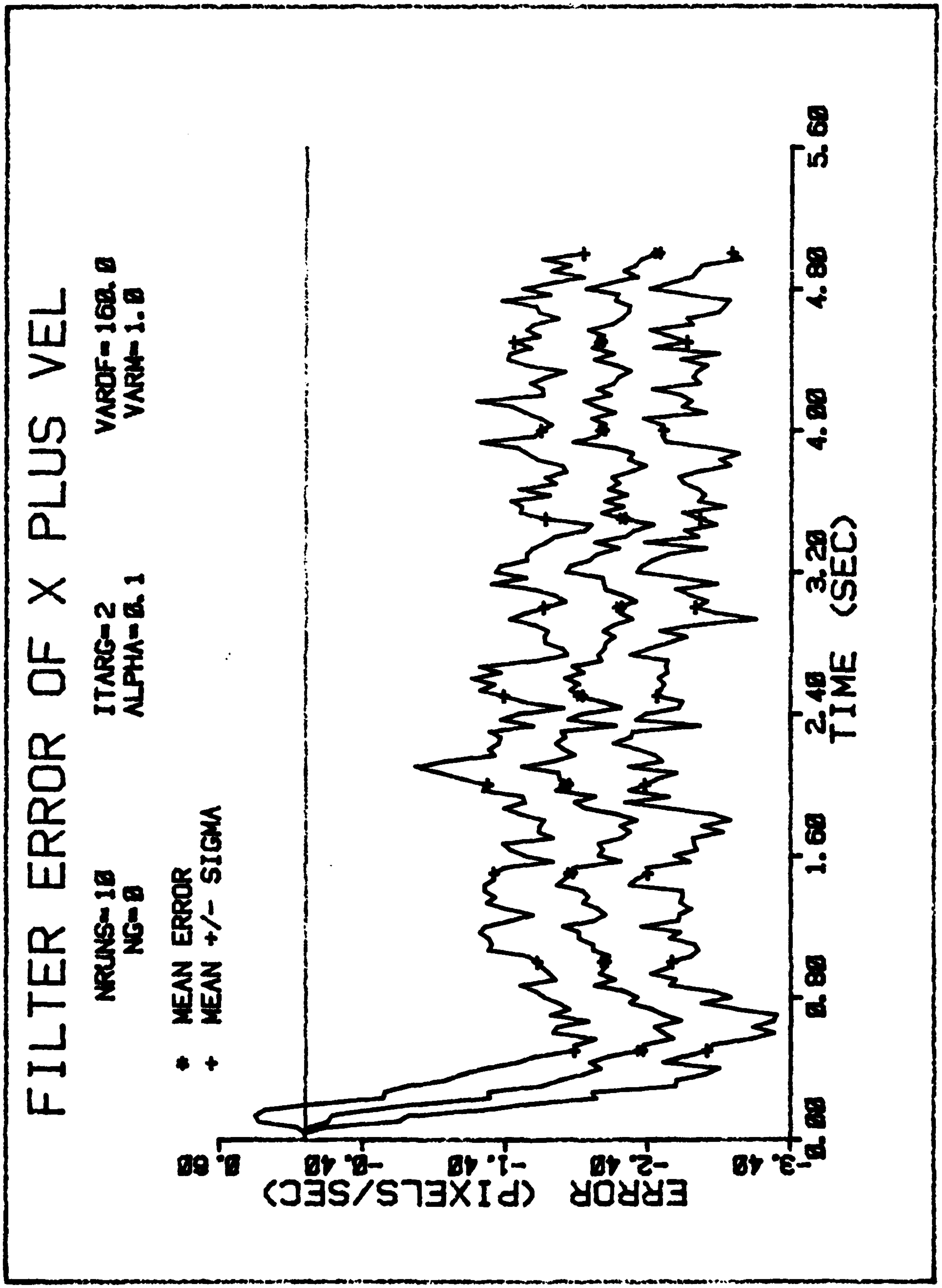


Figure E-5 Case 12 CTR Performance Plot

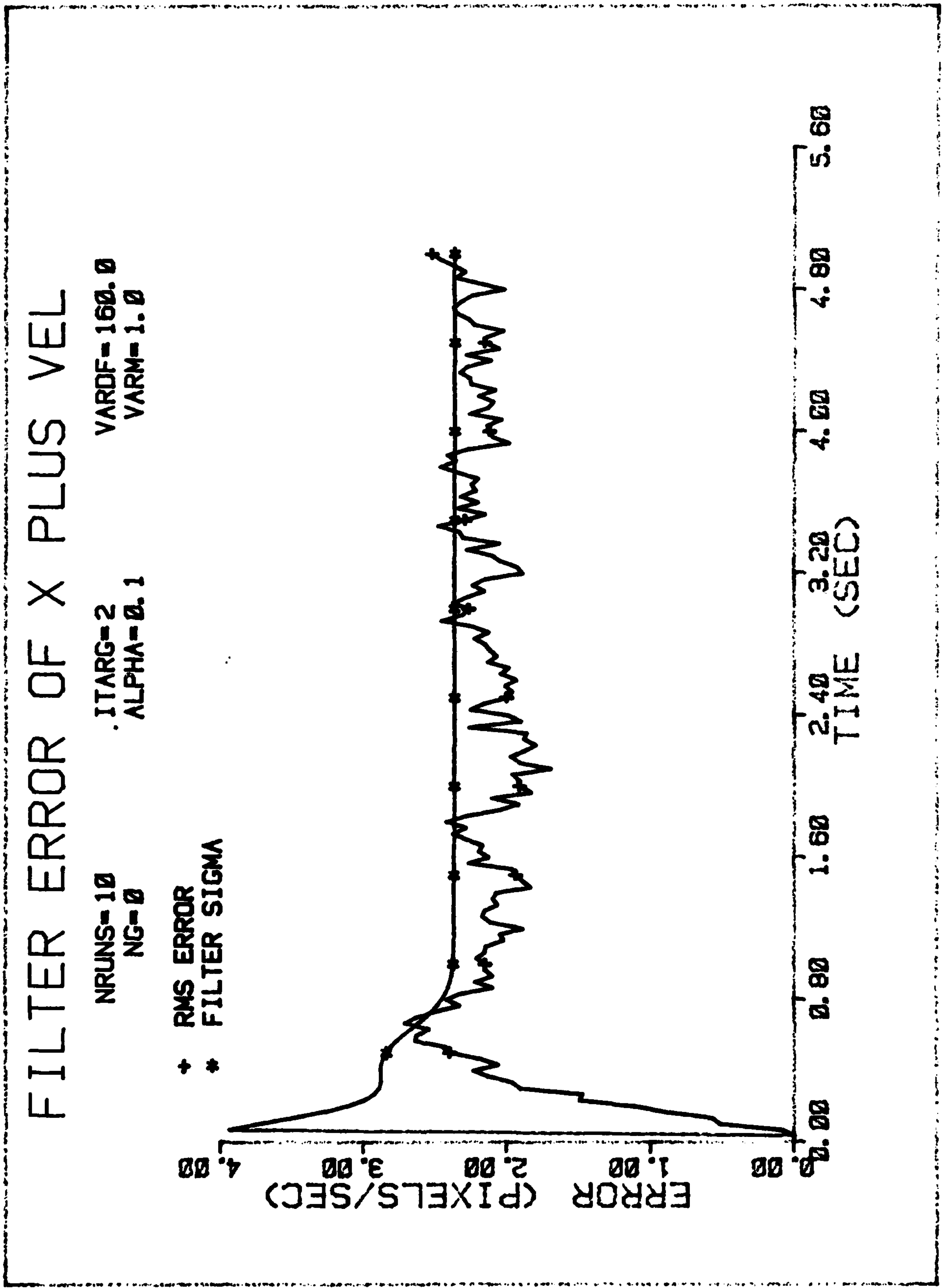


Figure E-6 Case 12 CTR Performance Plot

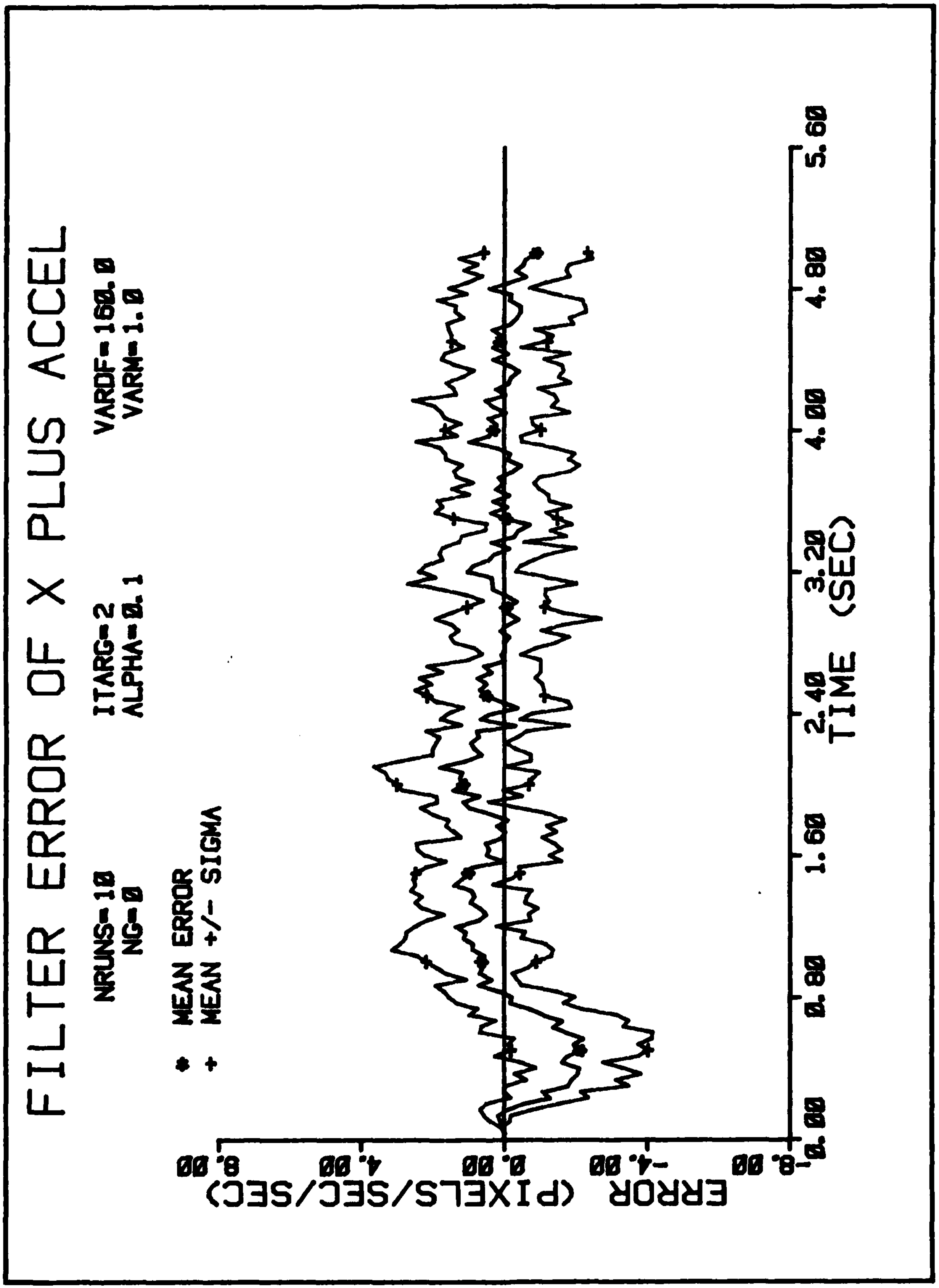


Figure E-7 Case 12 CTR Performance Plot

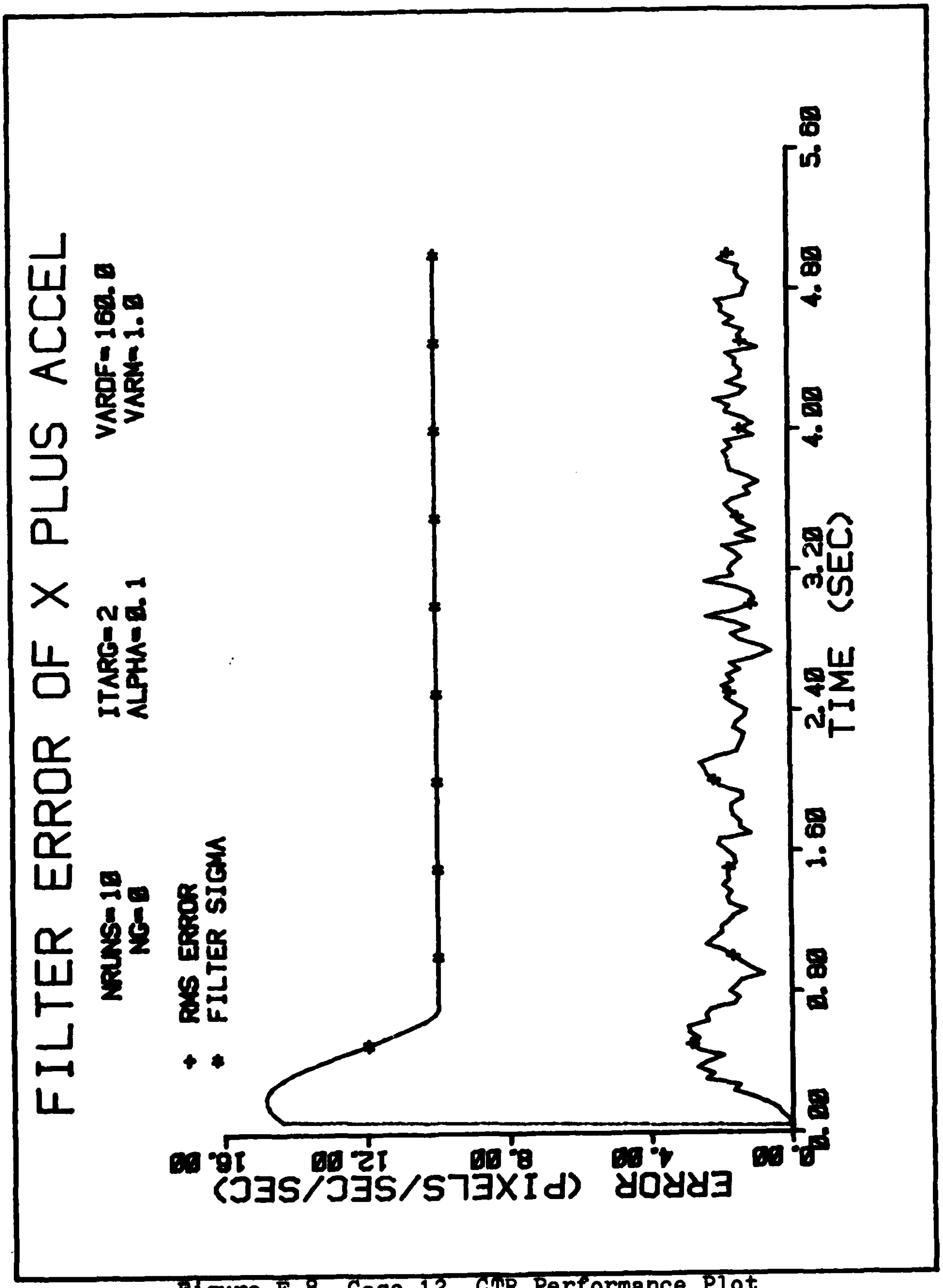


Figure E-8 Case 12 CTR Performance Plot

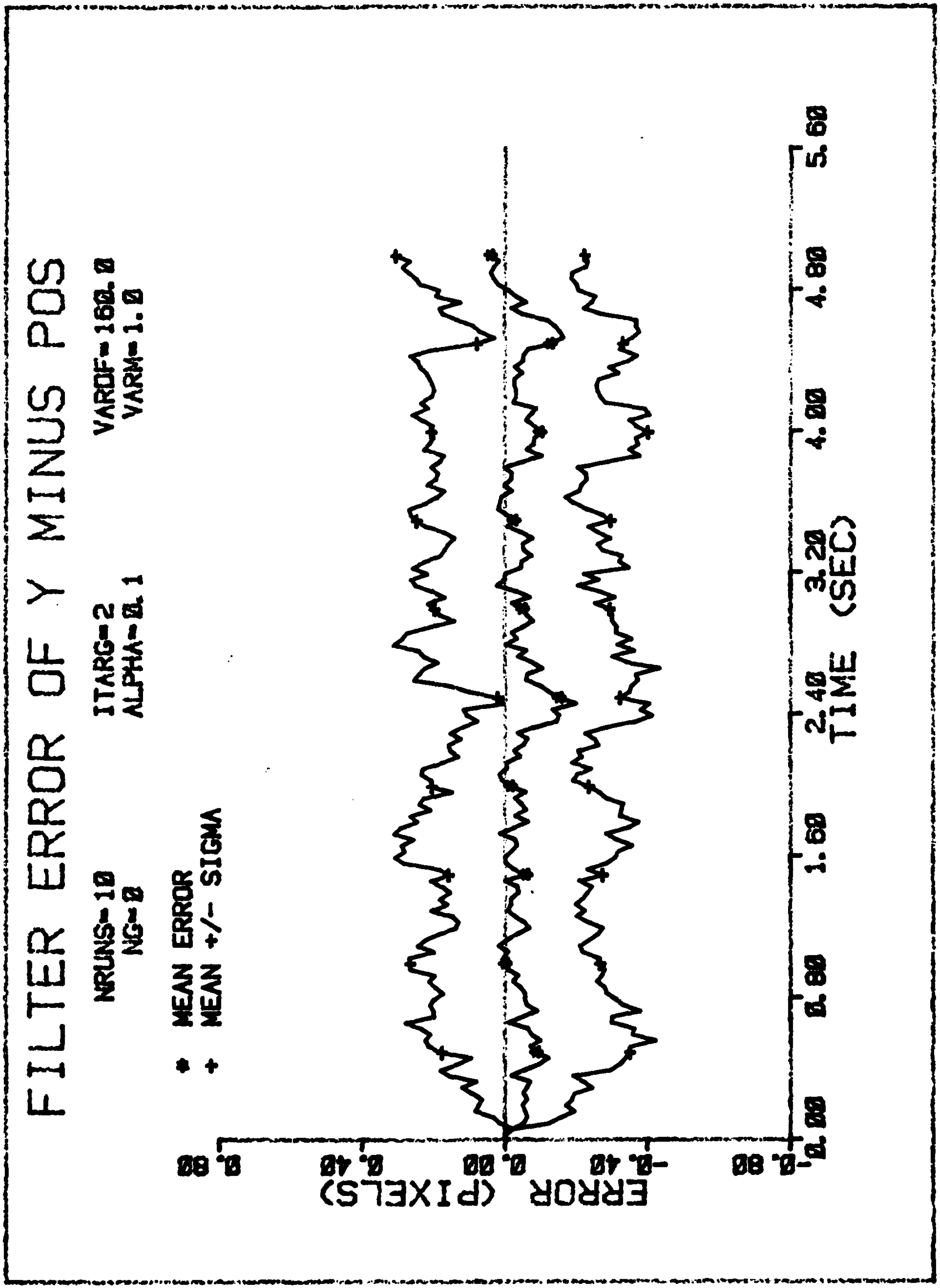


Figure E-9 Case 12 CTR Performance Plot

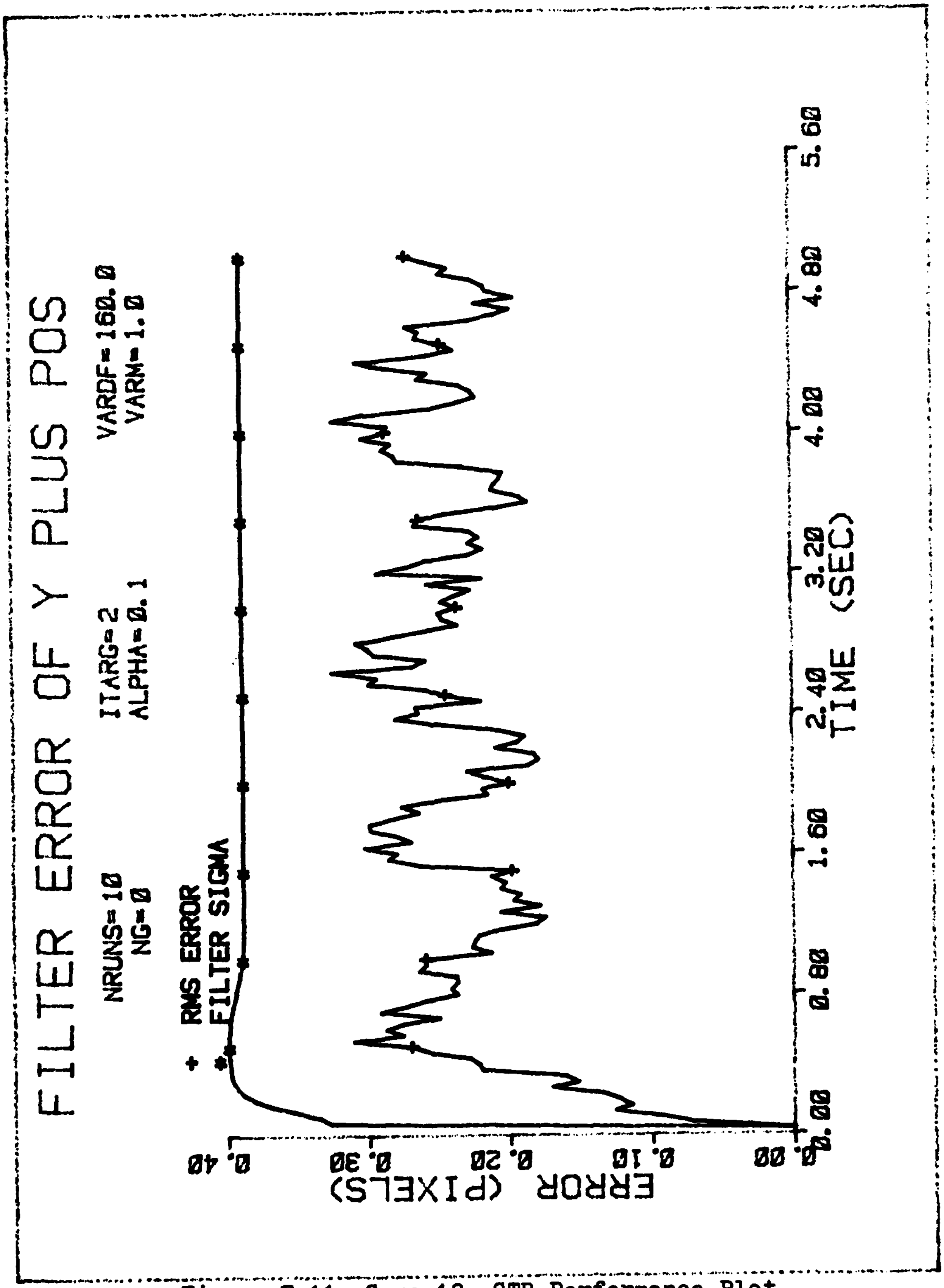


Figure E-11 Case 12 CTR Performance Plot

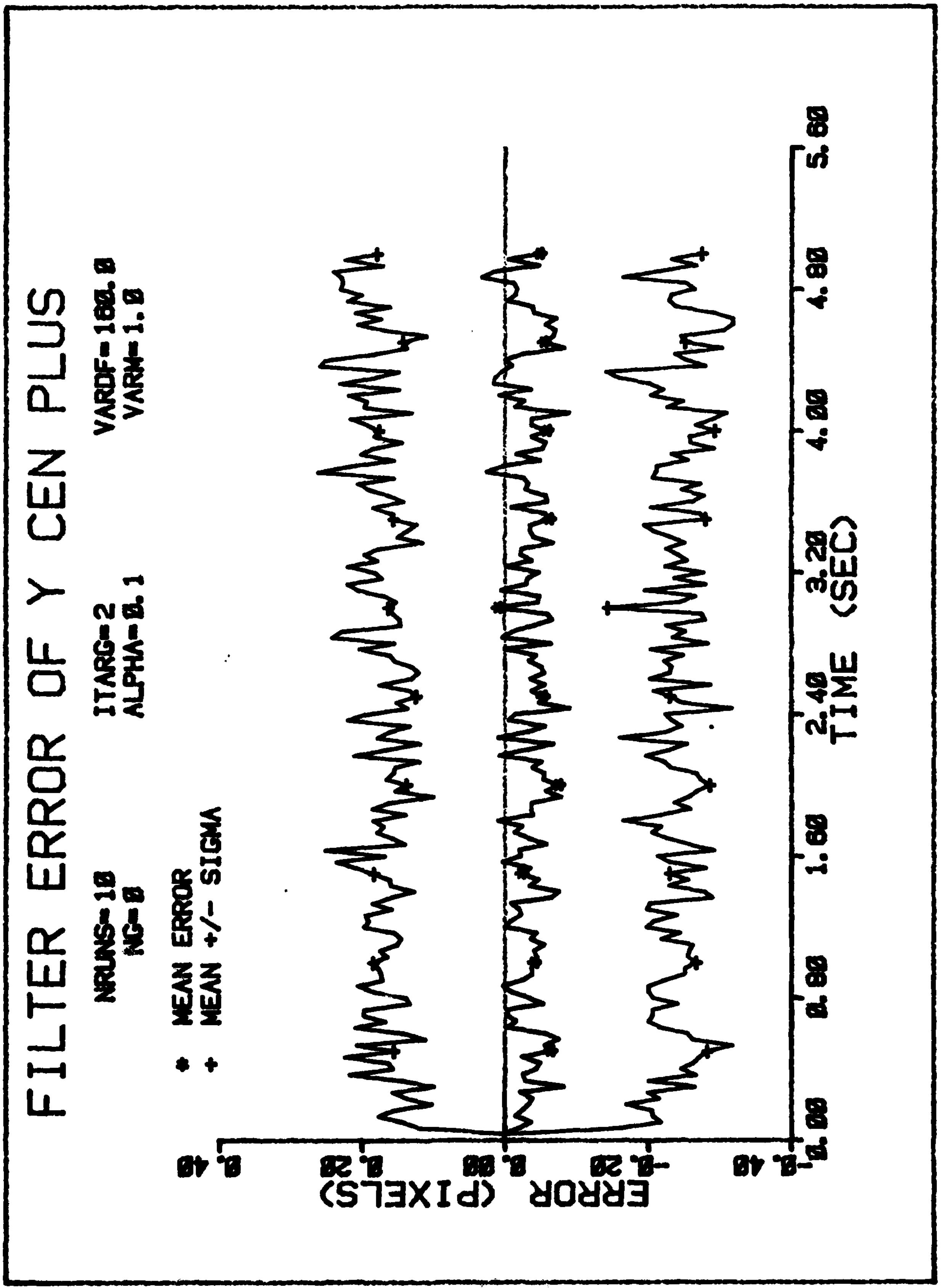


Figure E-12 Case 12 CTR Performance Plot

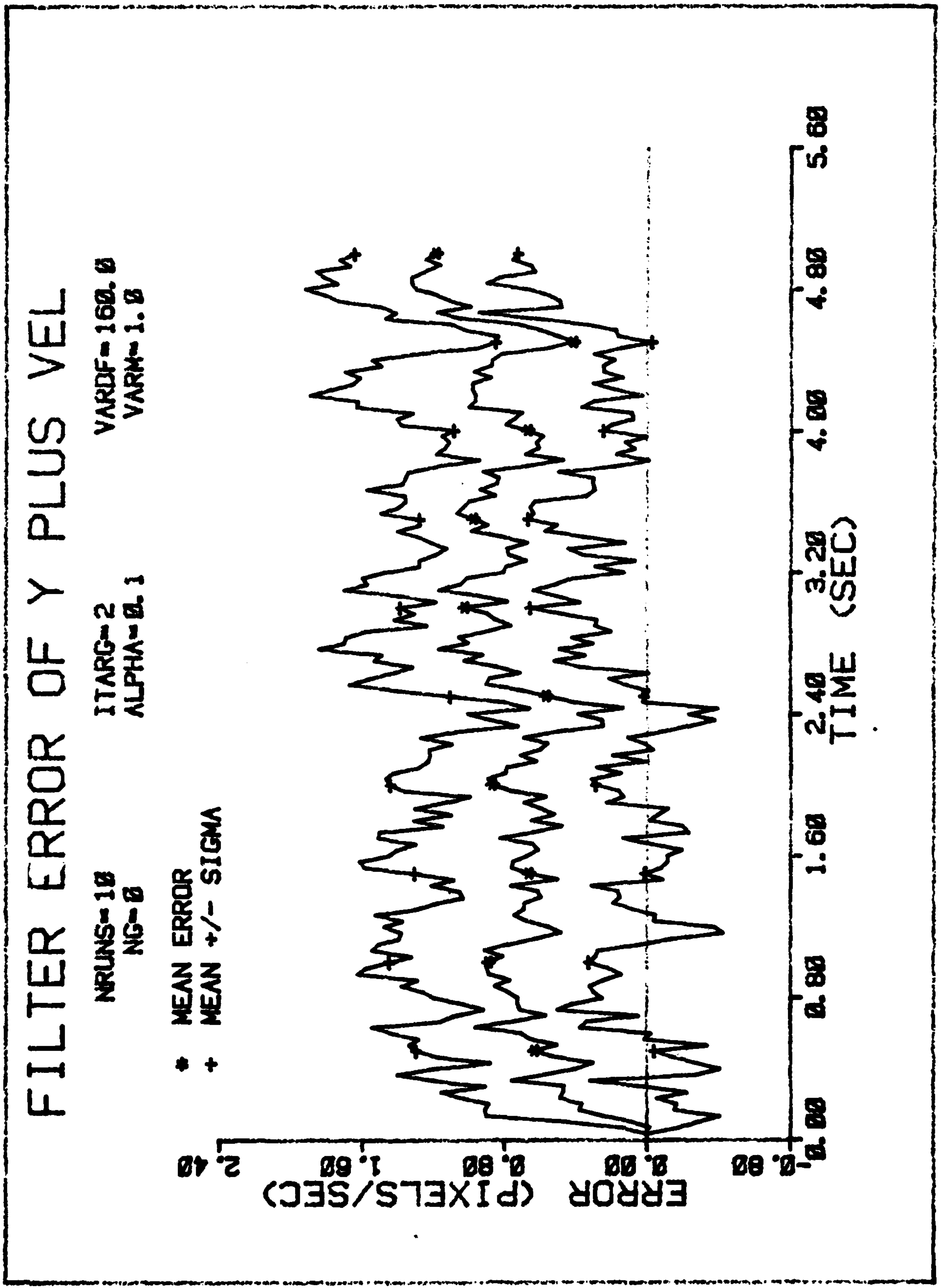


Figure E-13 Case 12 CTR Performance Plot

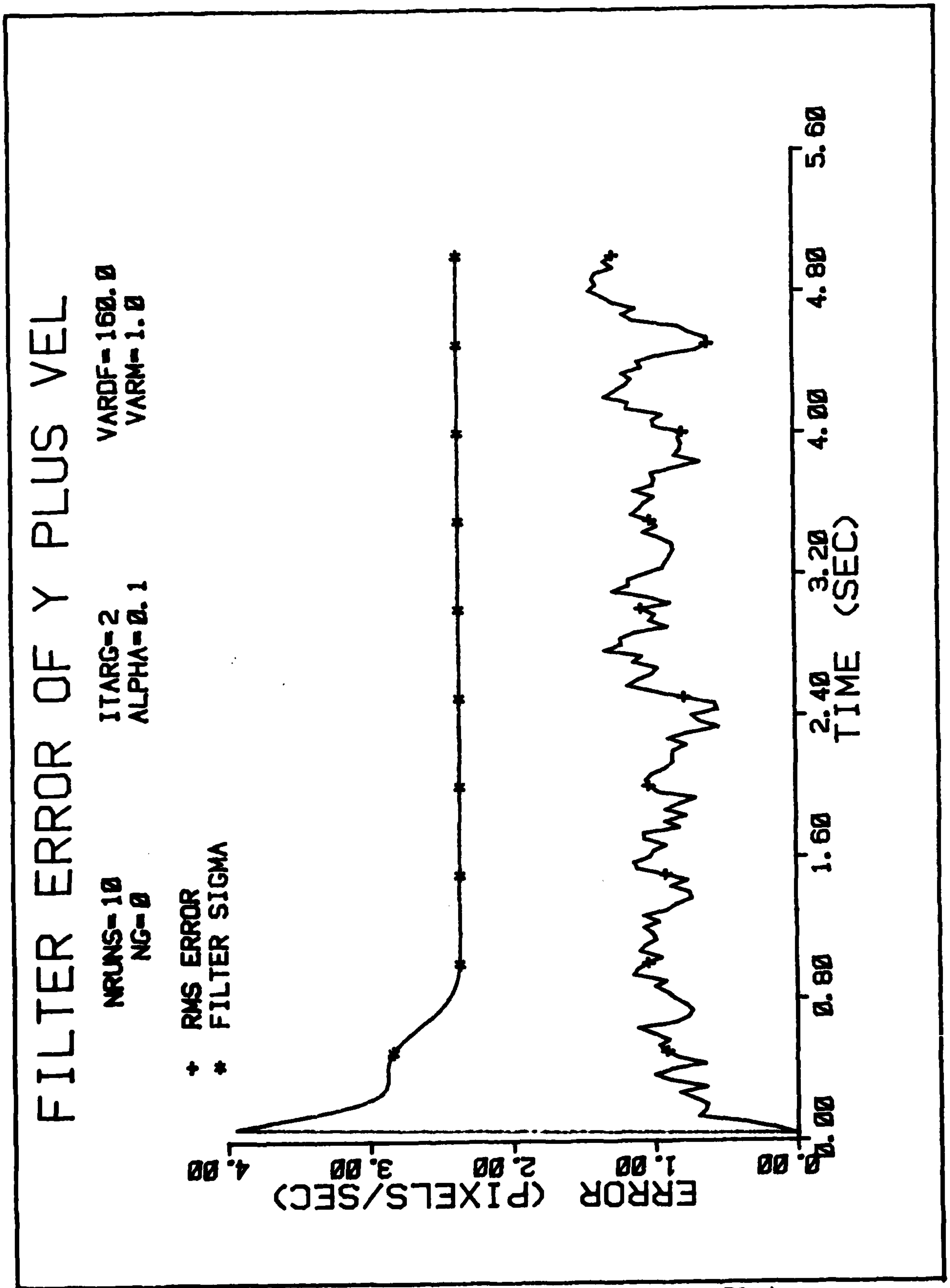


Figure E-14 Case 12 CTR Performance Plot

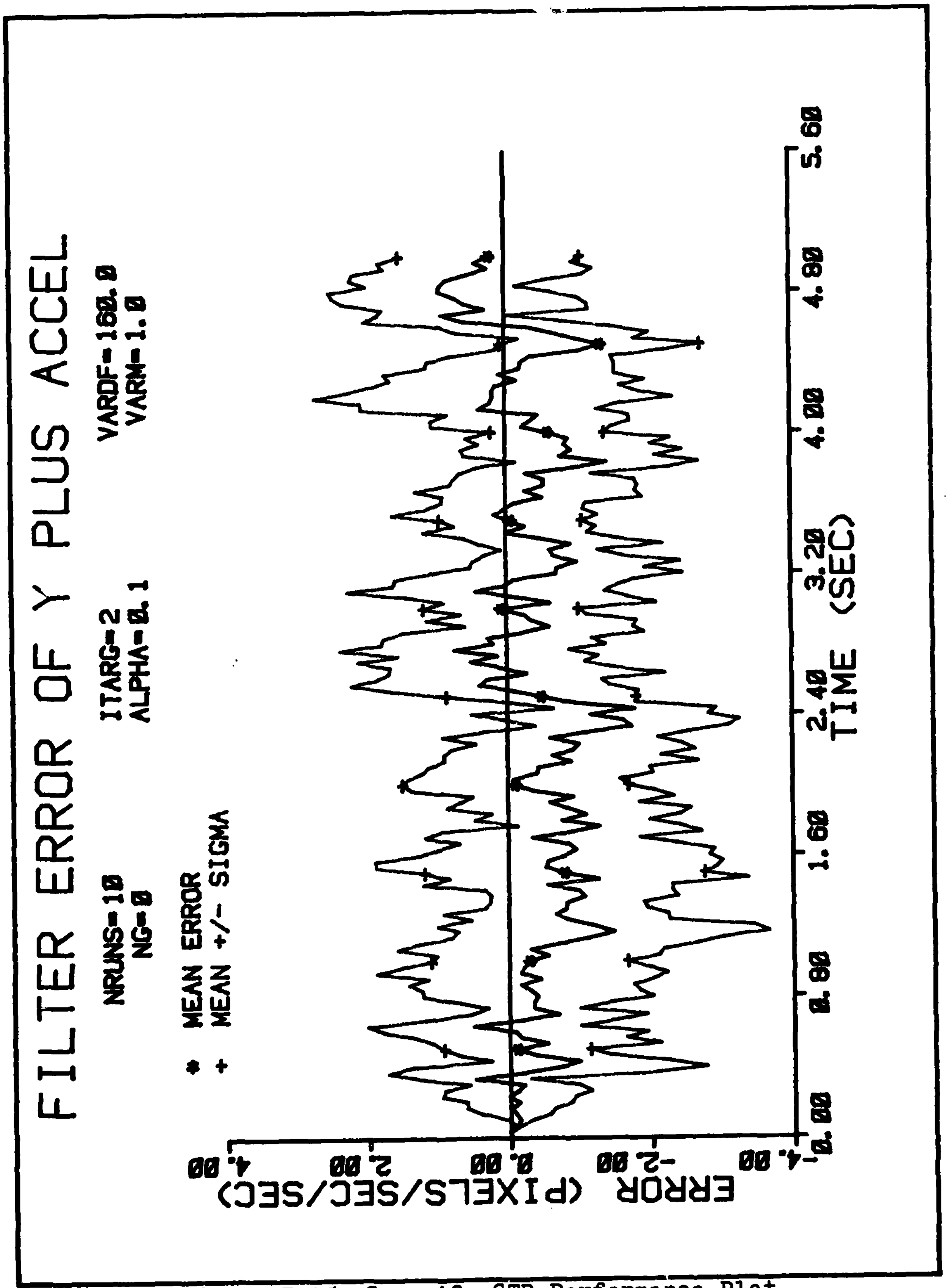


Figure E-15 Case 12 CTR Performance Plot

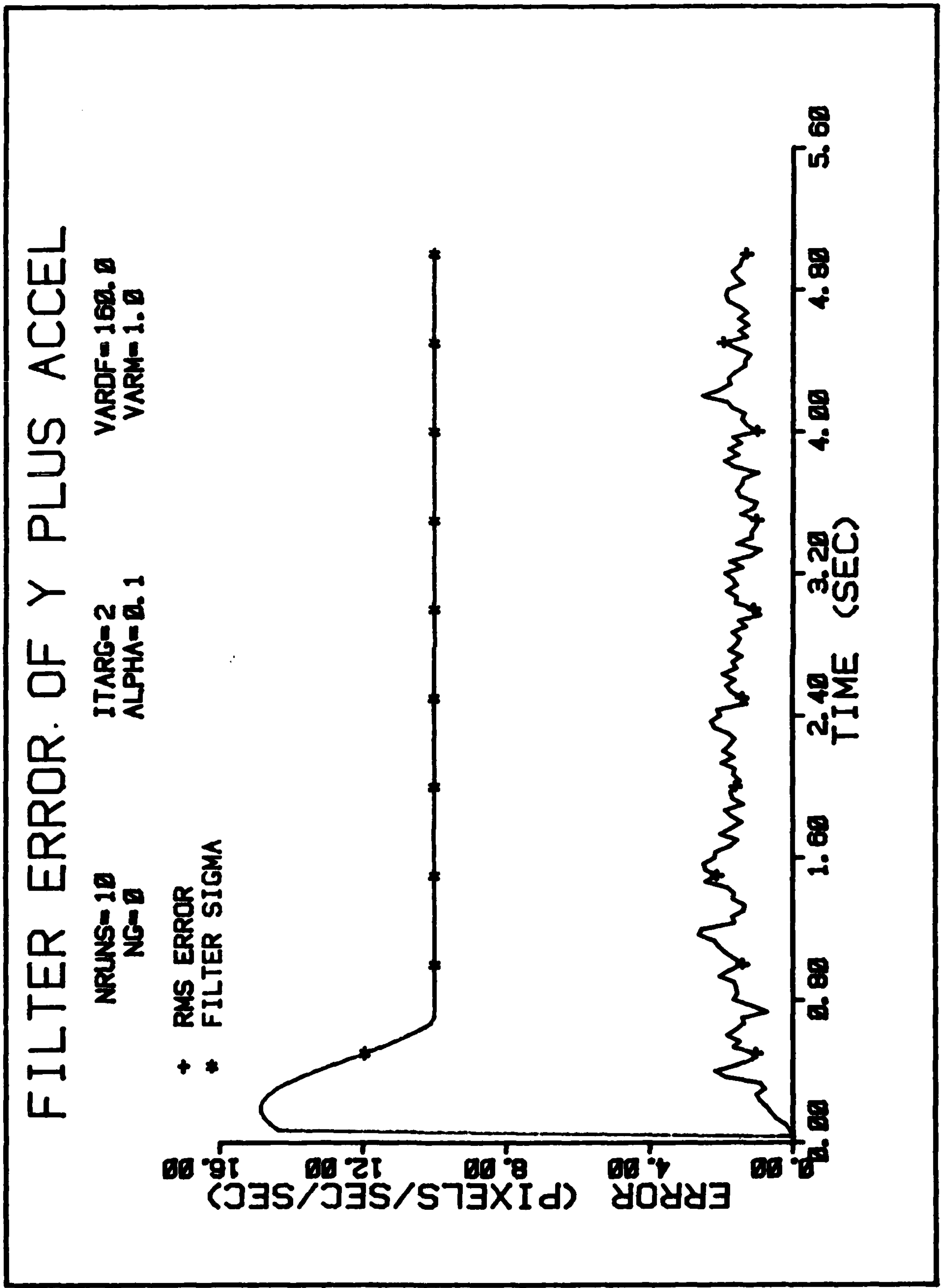


Figure E-16 Case 12 CTR Performance Plot

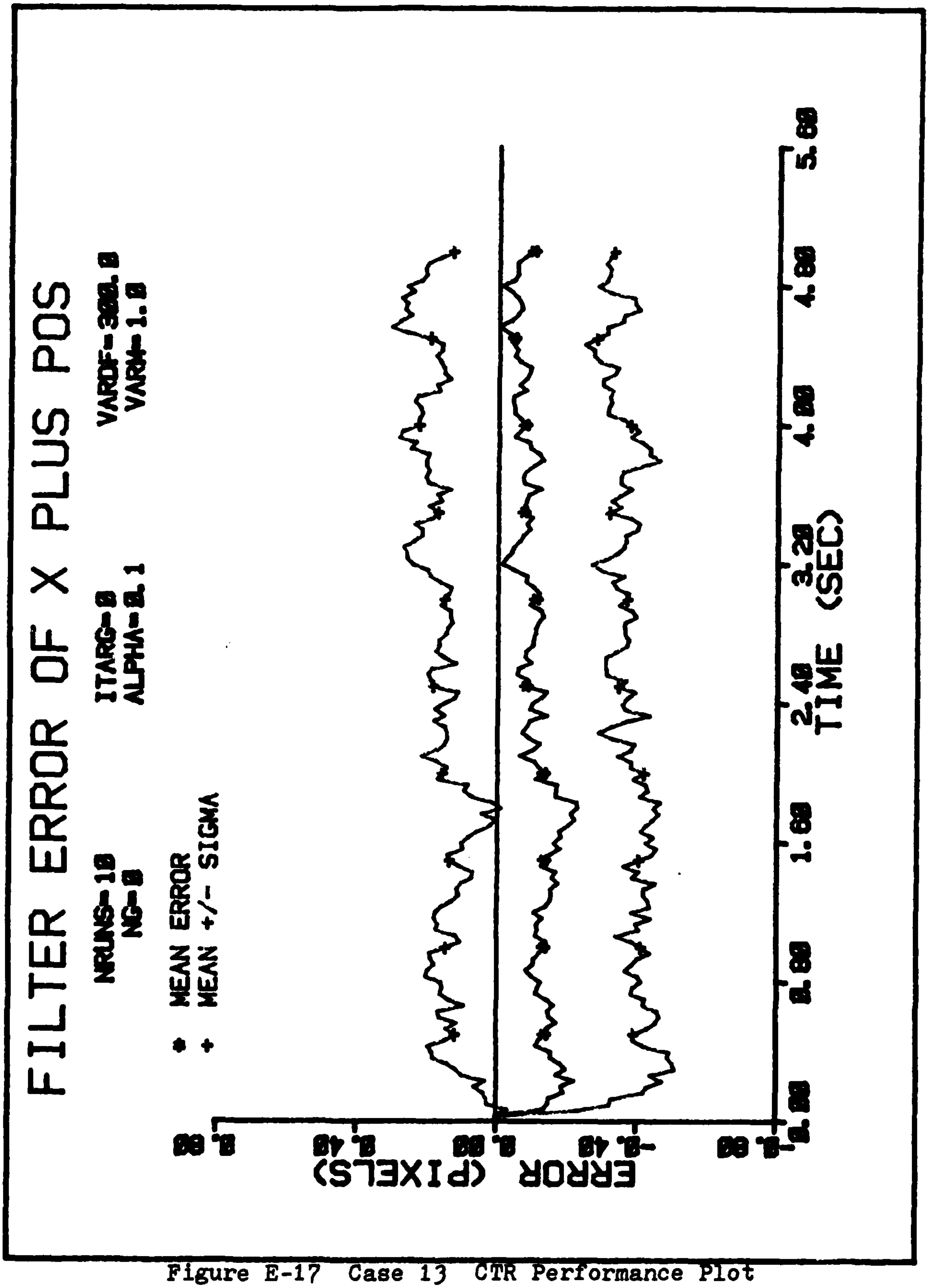


Figure E-17 Case 13 CTR Performance Plot

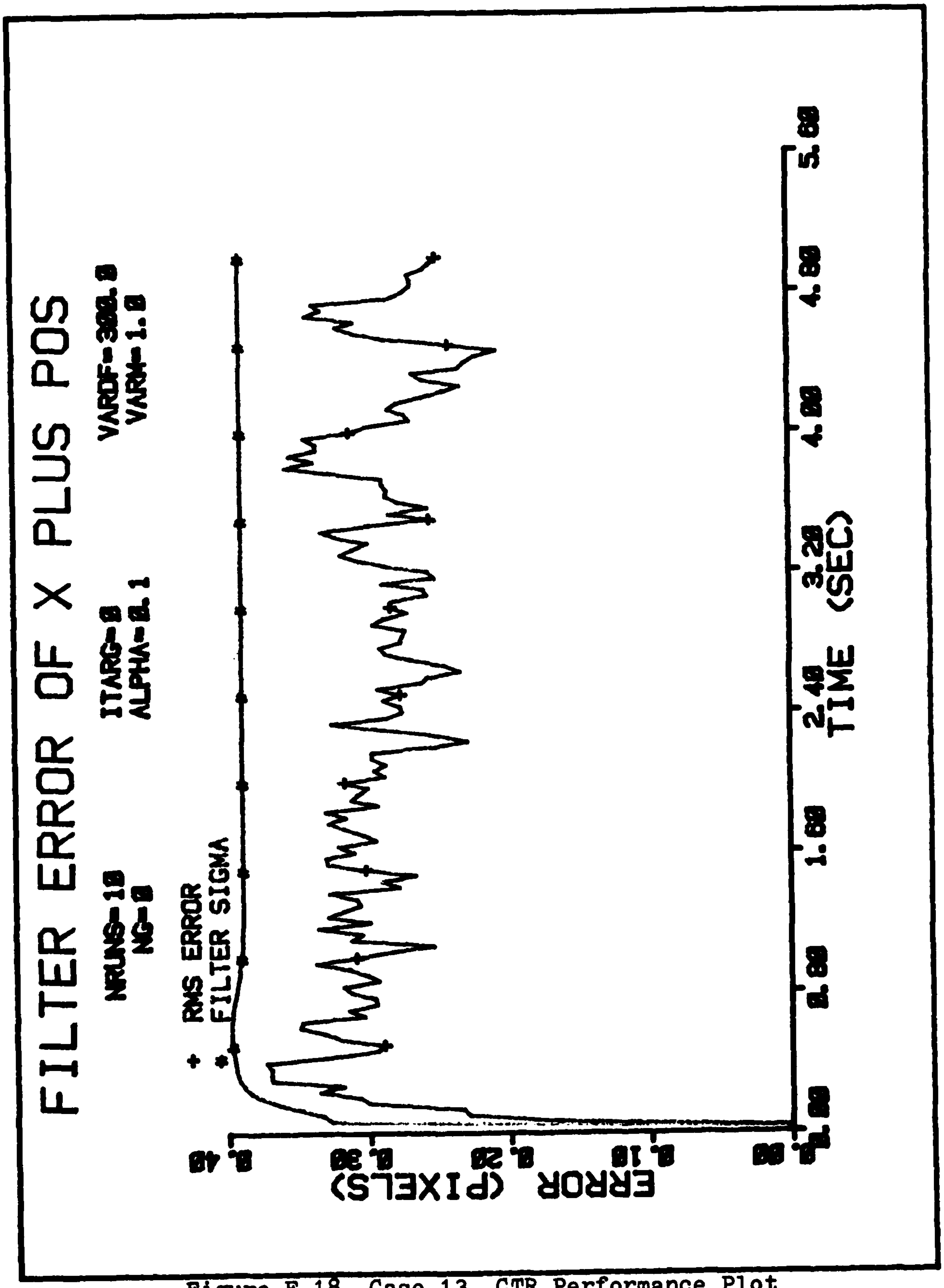


Figure E-18 Case 13 CTR Performance Plot

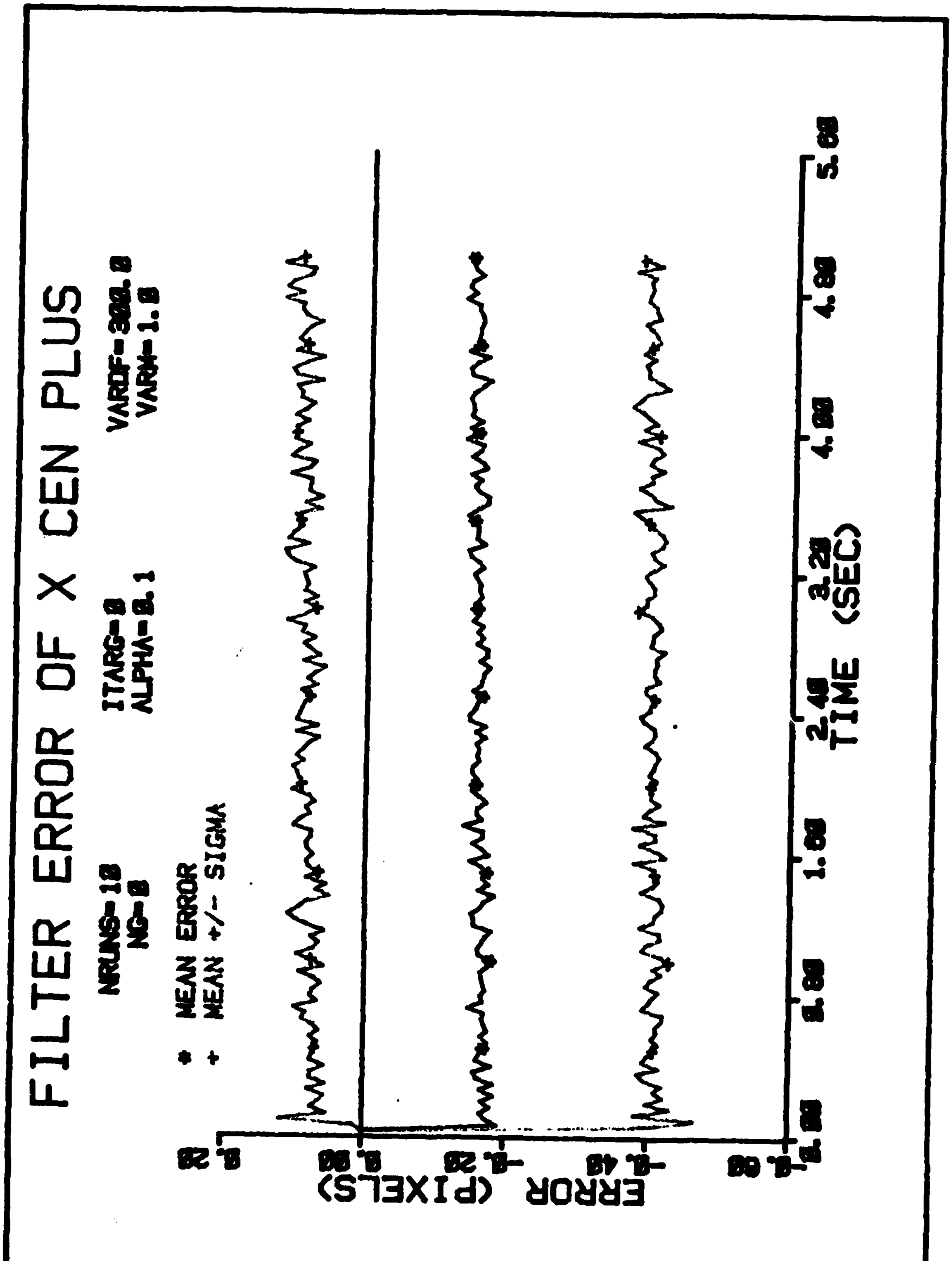


Figure E-19 Case 13 CTR Performance Plot

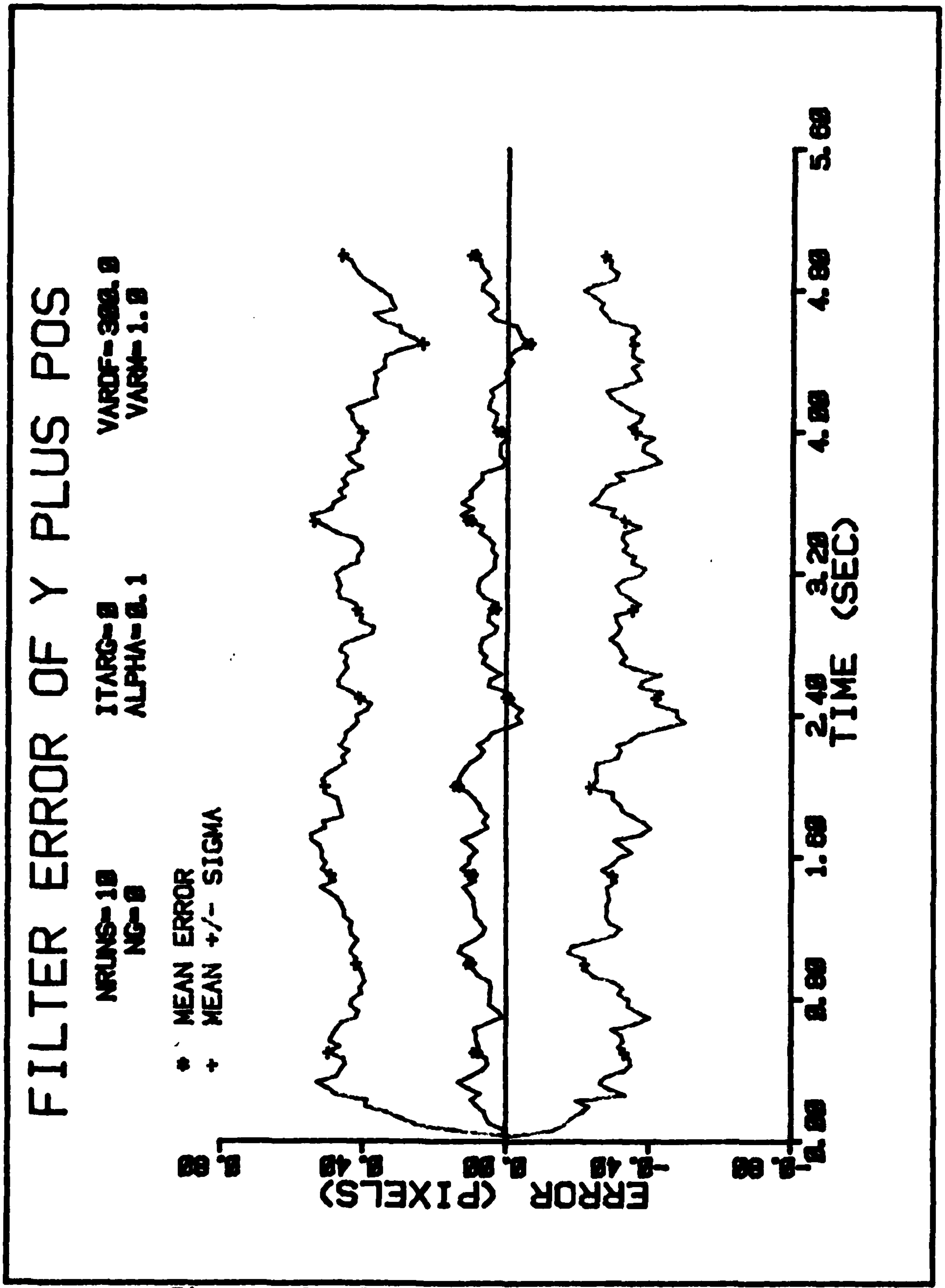


Figure E-20 Case 13 CTR Performance Plot

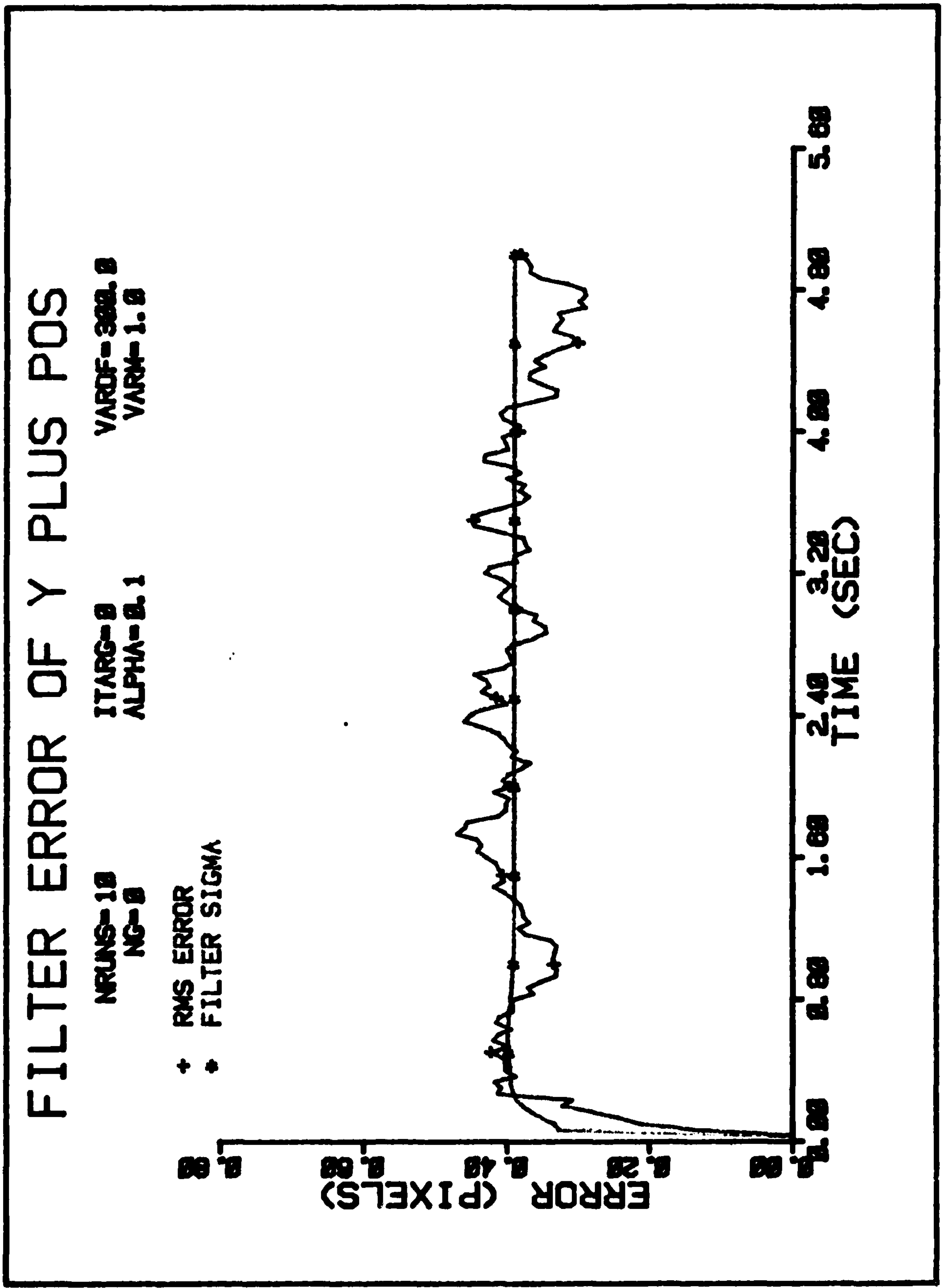


Figure E-21 Case 13 CTR Performance Plot

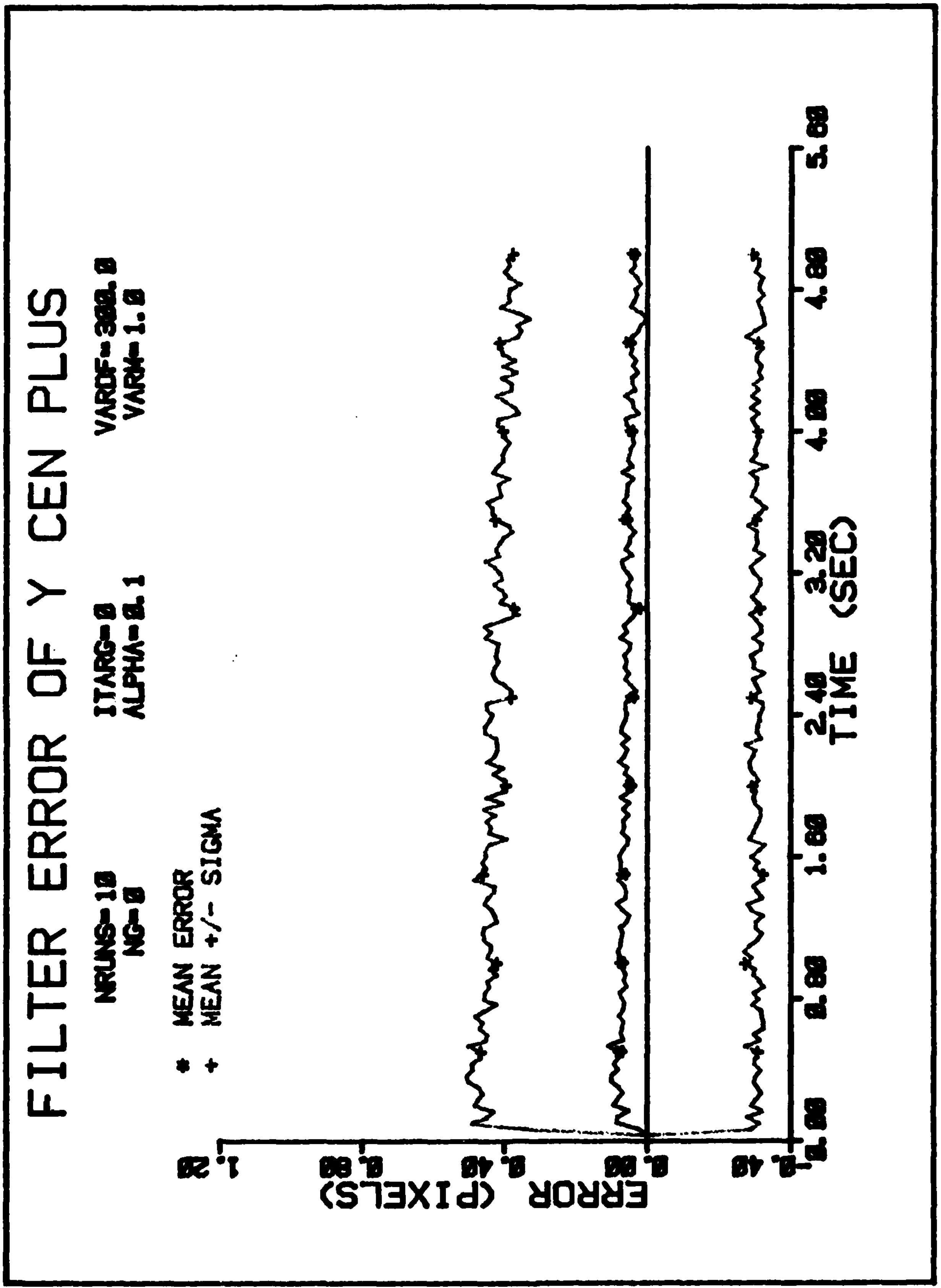


Figure E-22 Case 13 CTR Performance Plot

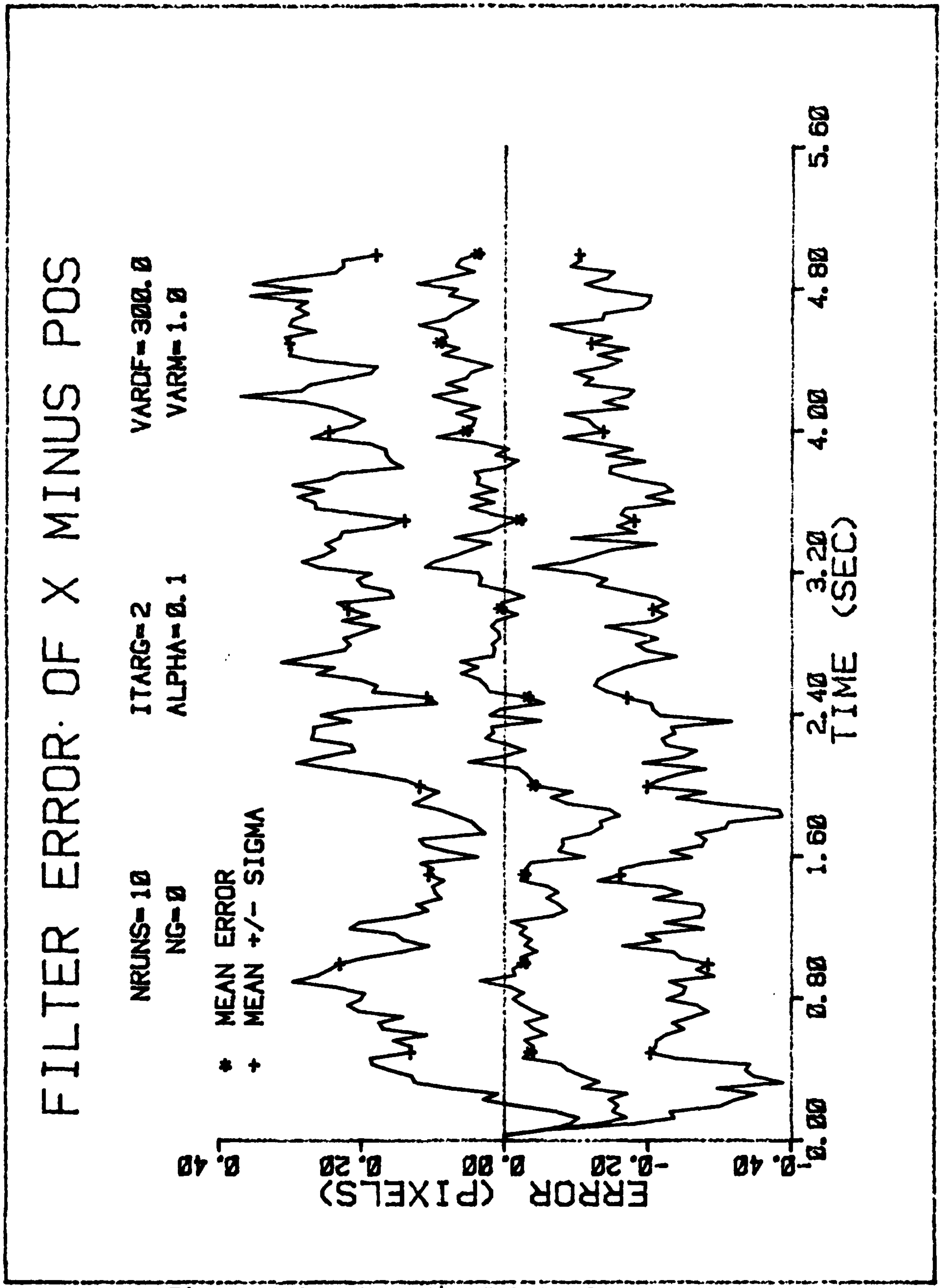


Figure E-23 Case 14 CTR Performance Plot

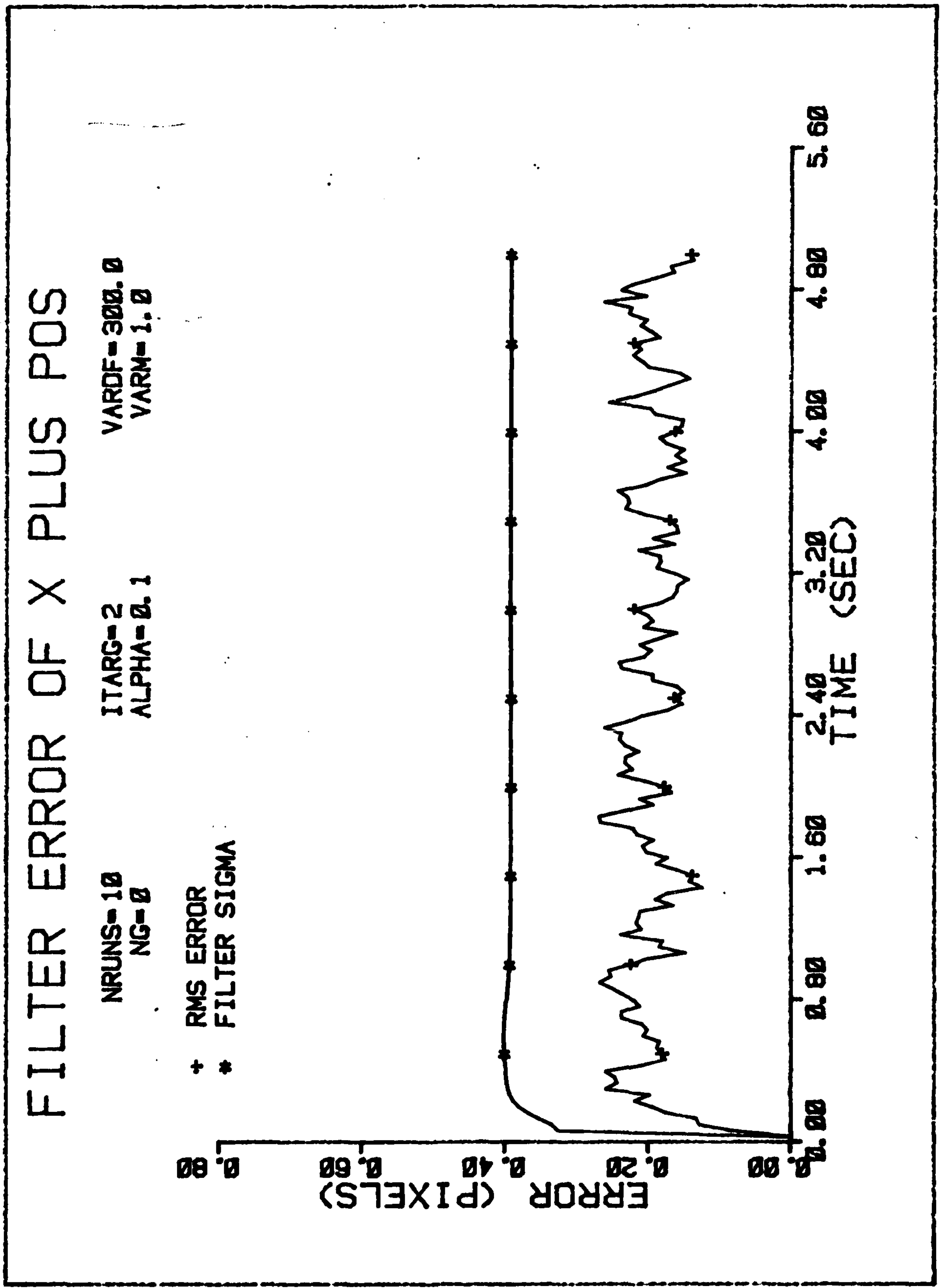


Figure E-25 Case 14 .CTR Performance Plot

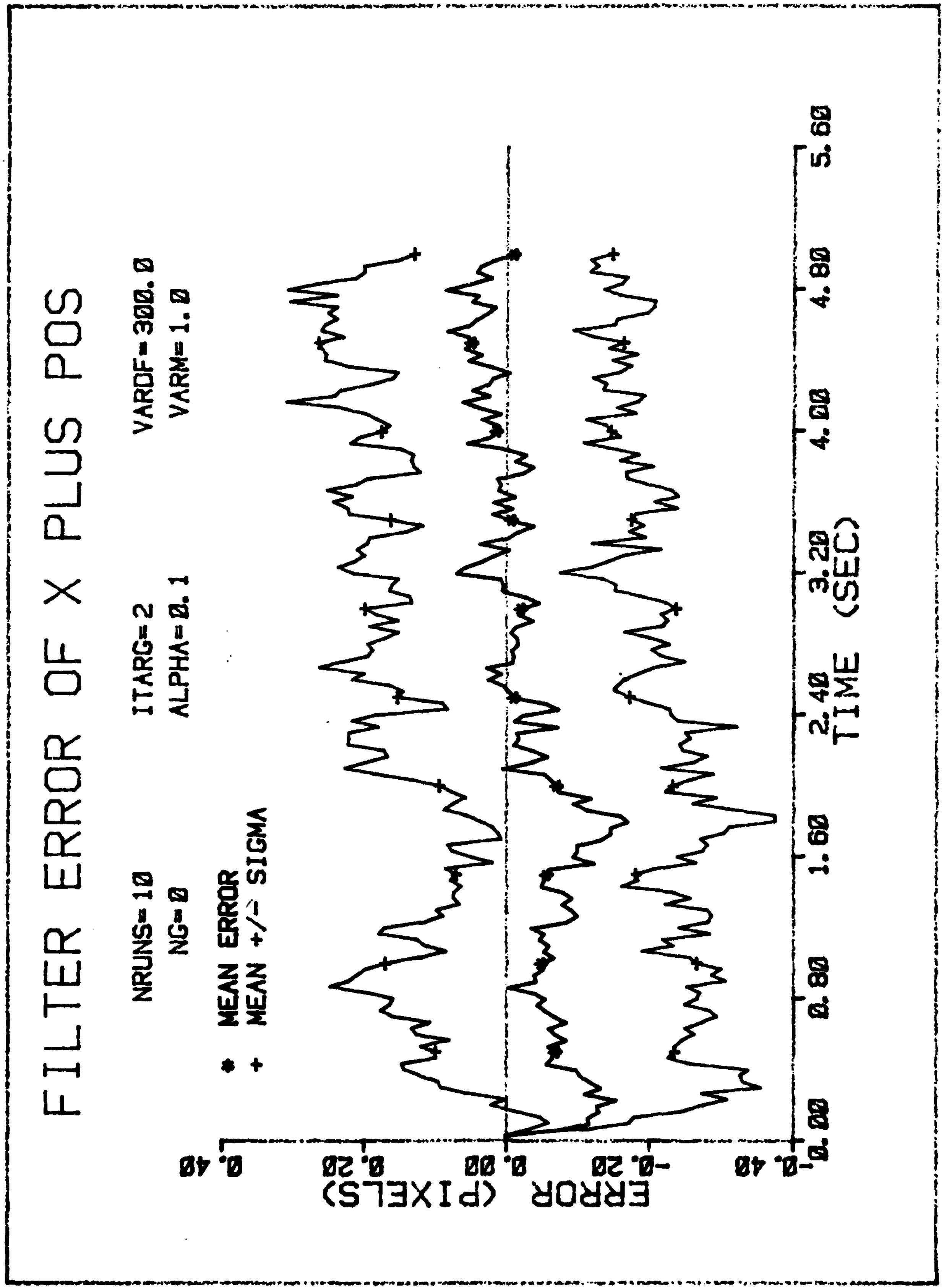


Figure E-24 Case 14 CTR Performance Plot

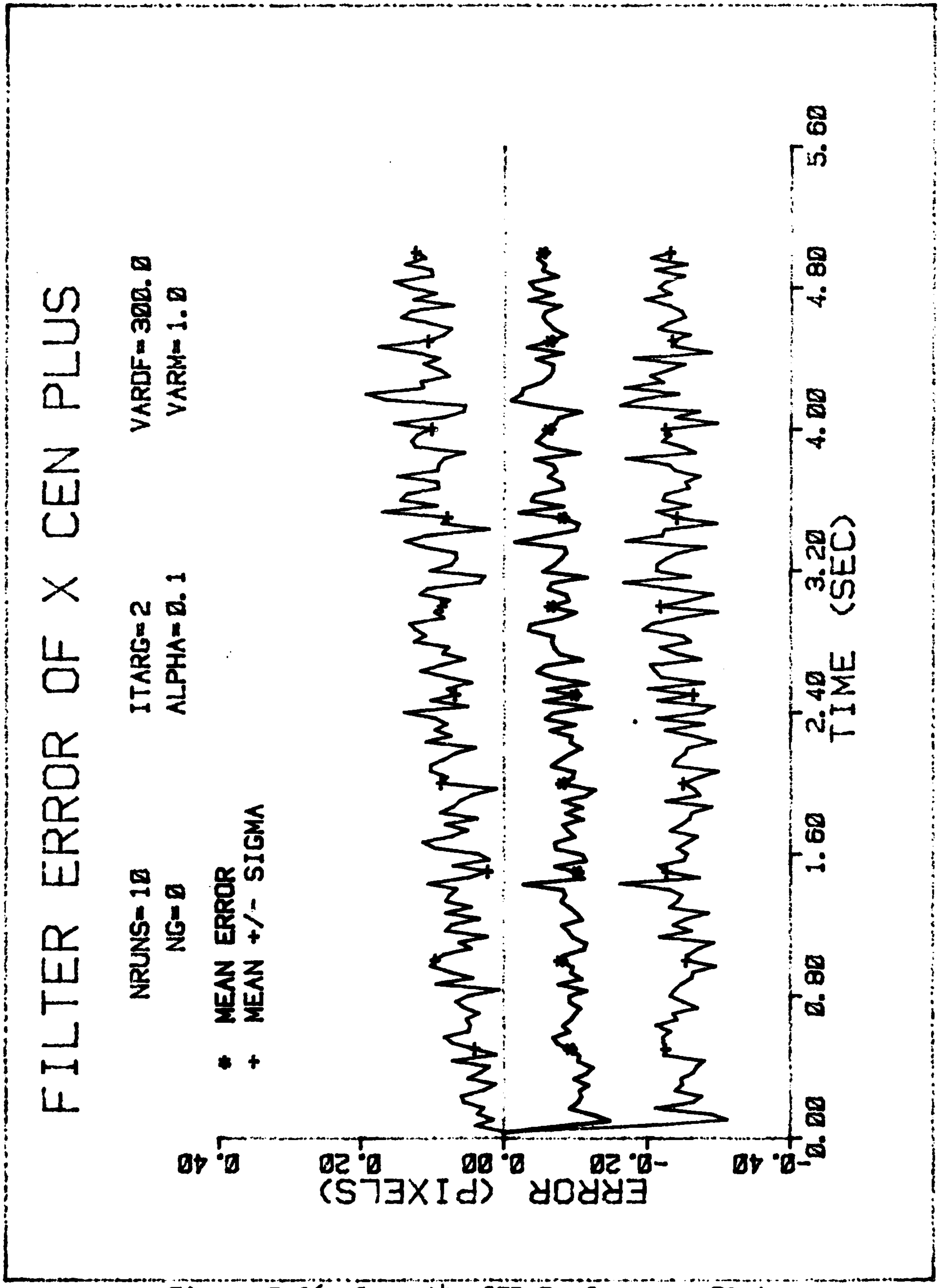


Figure E-26 Case 14 CTR Performance Plot

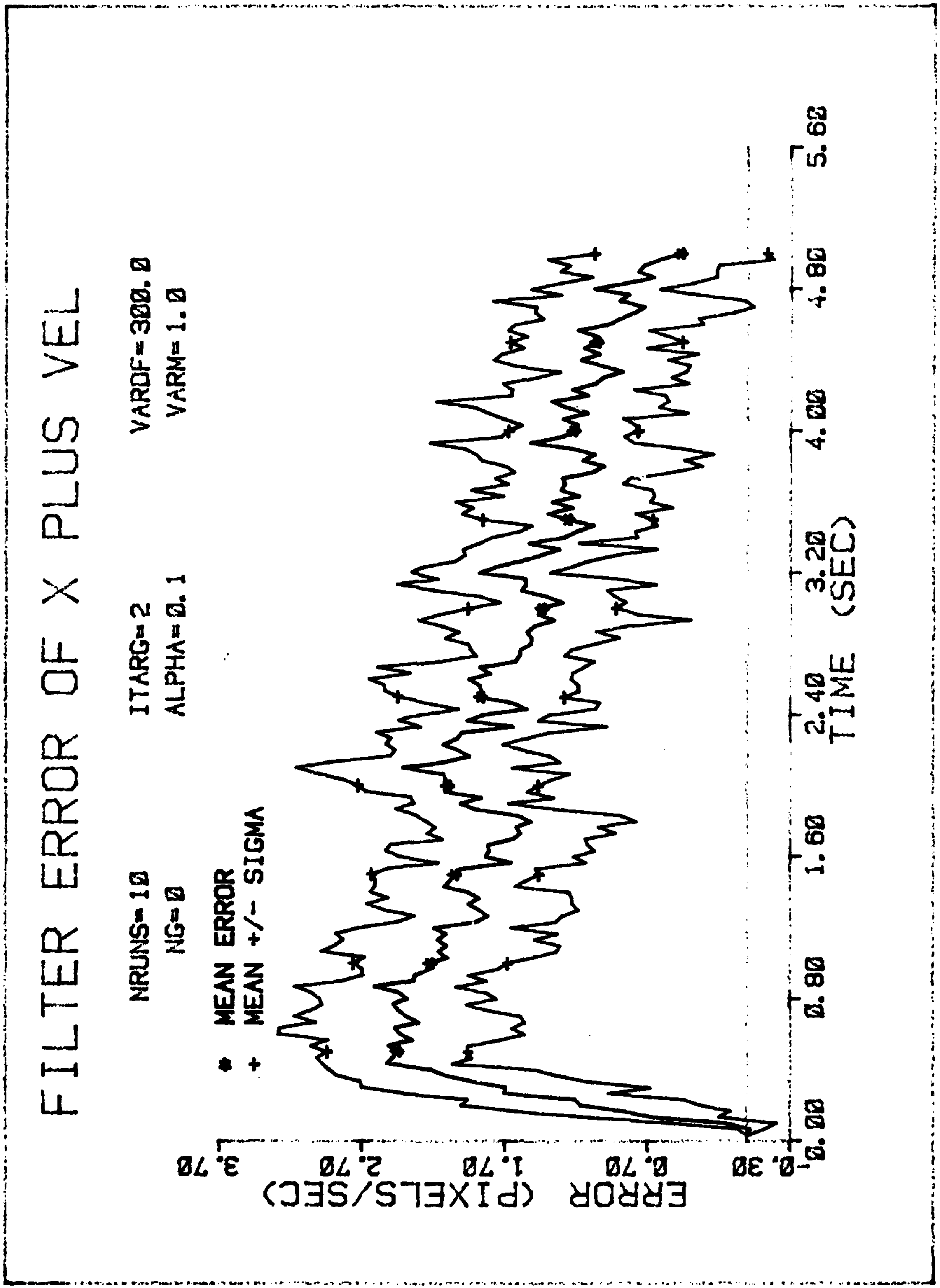


Figure E-27 Case 14 CTR Performance Plot

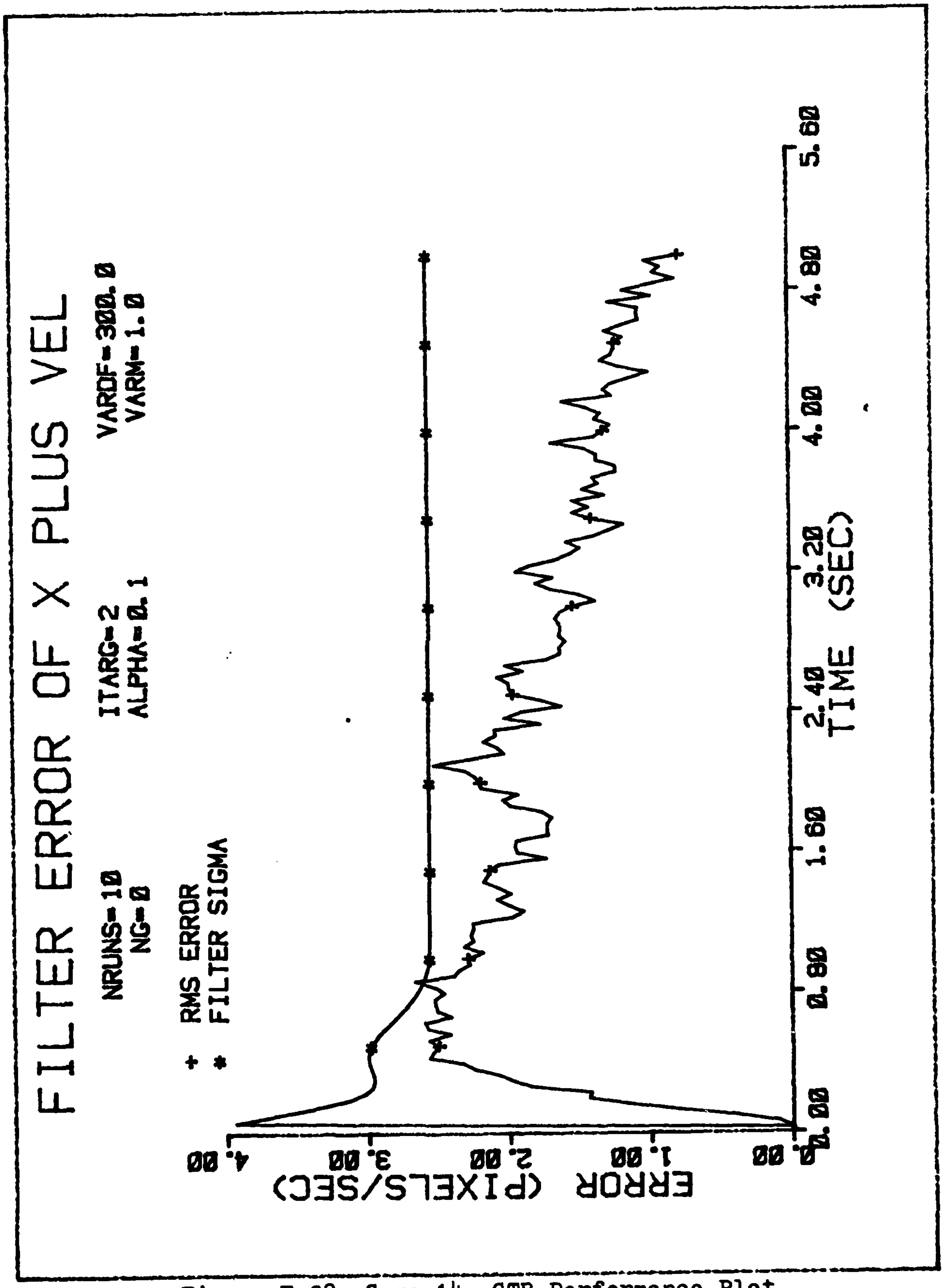


Figure E-28 Case 14 CTR Performance Plot

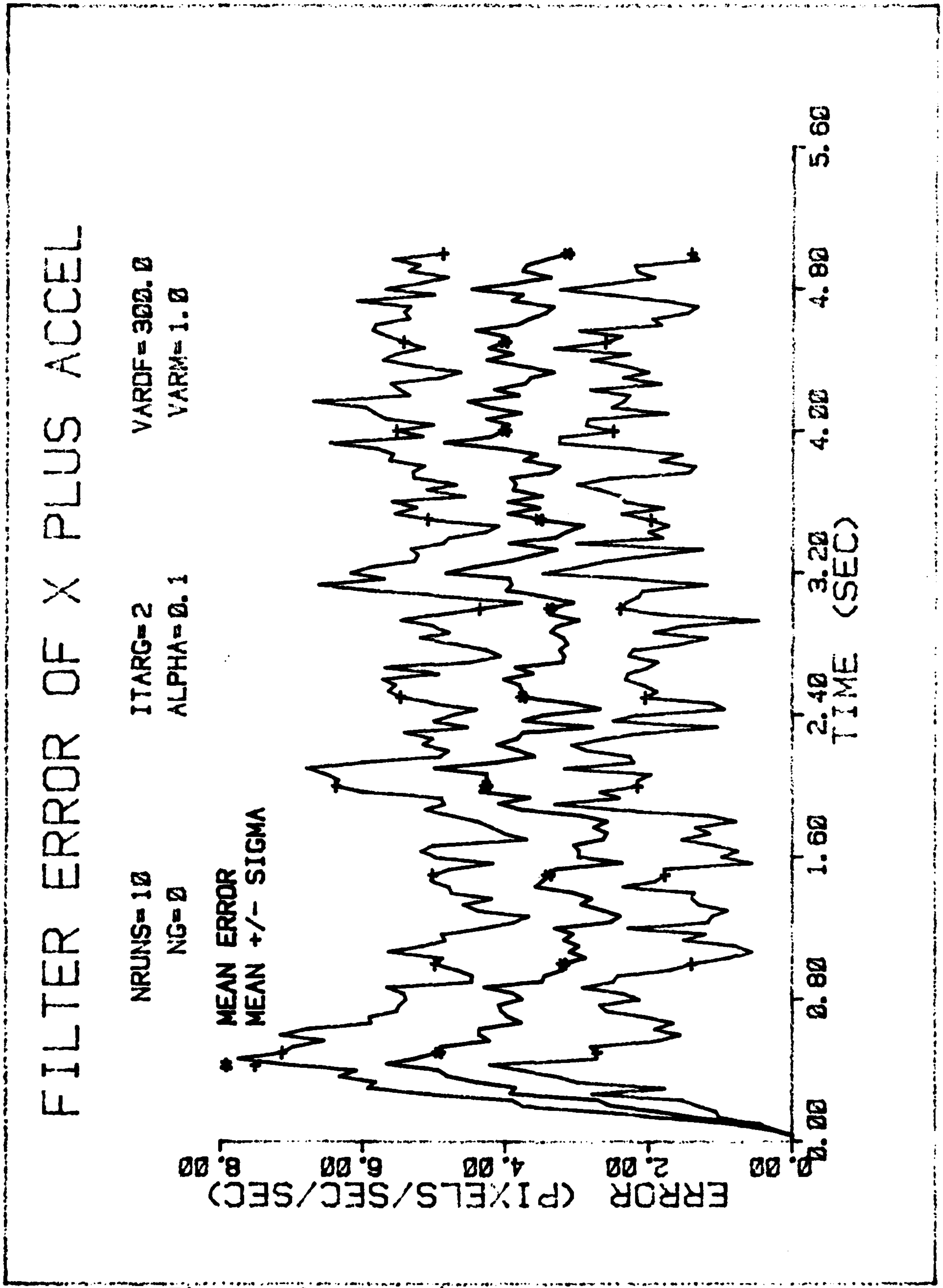


Figure E-29 Case 14 CTR Performance Plot

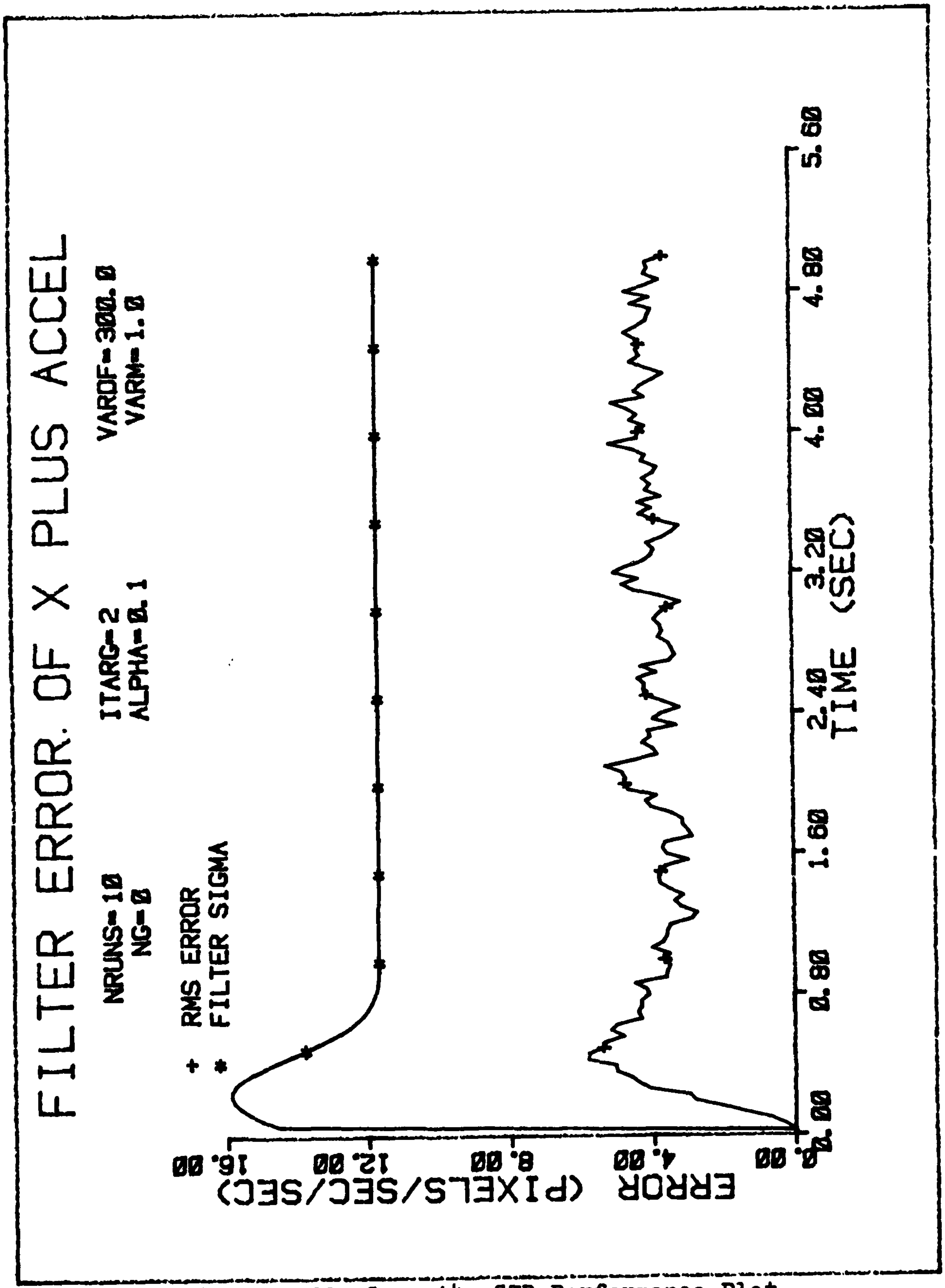


Figure E-30 Case 14 CTR Performance Plot

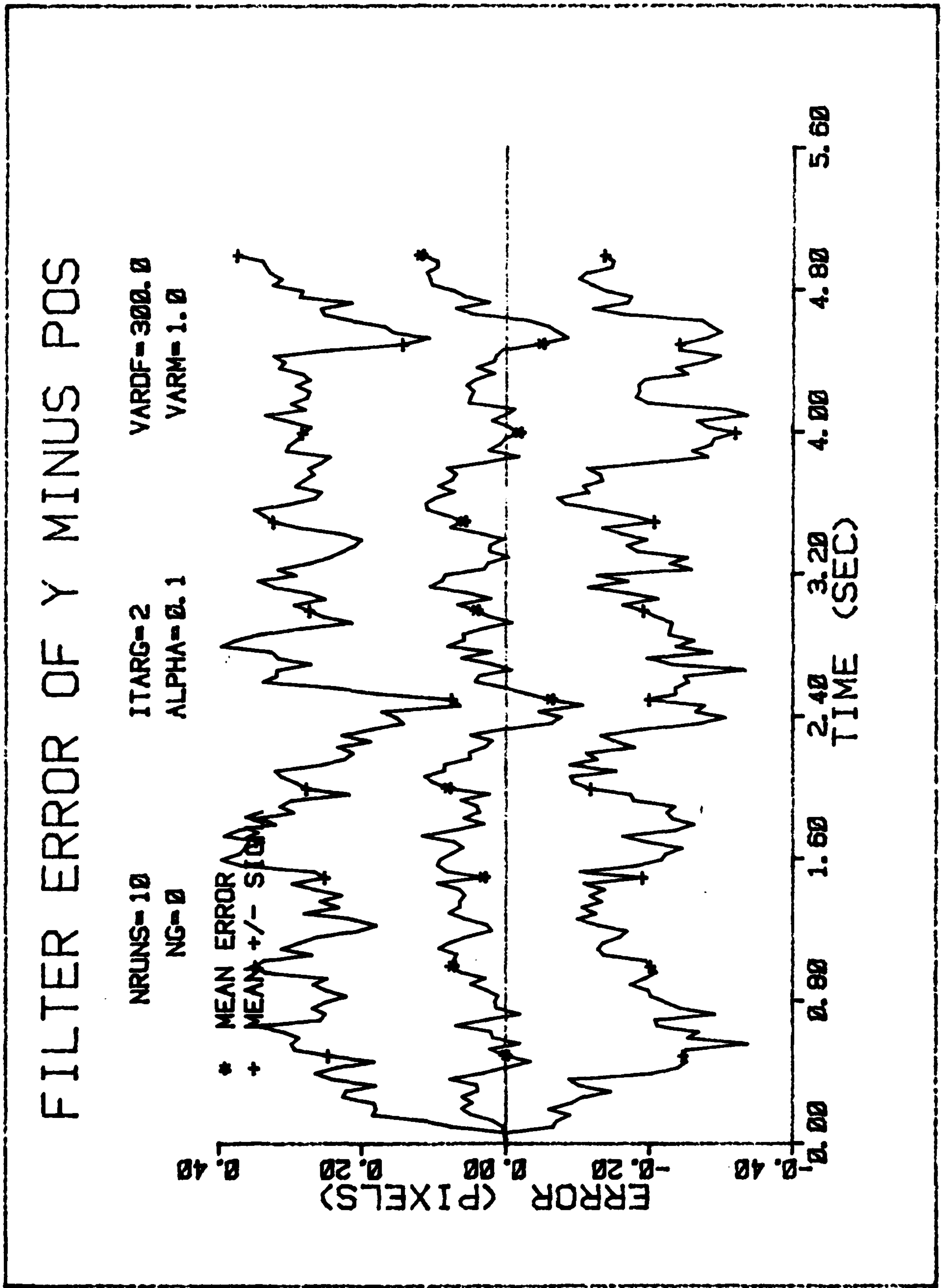


Figure E-31 Case 14 CTR Performance Plot

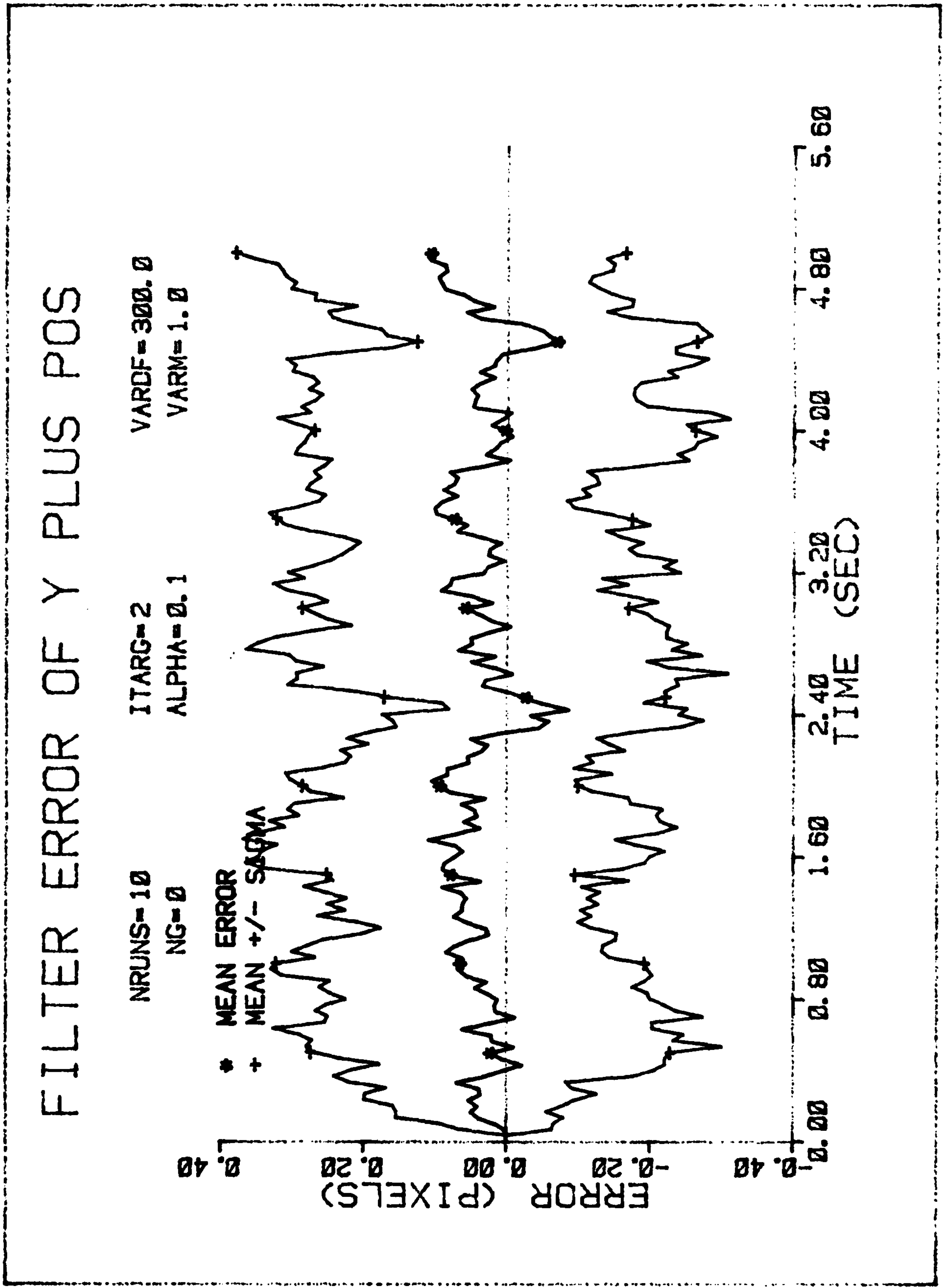


Figure E-32 Case 14 CTR Performance Plot

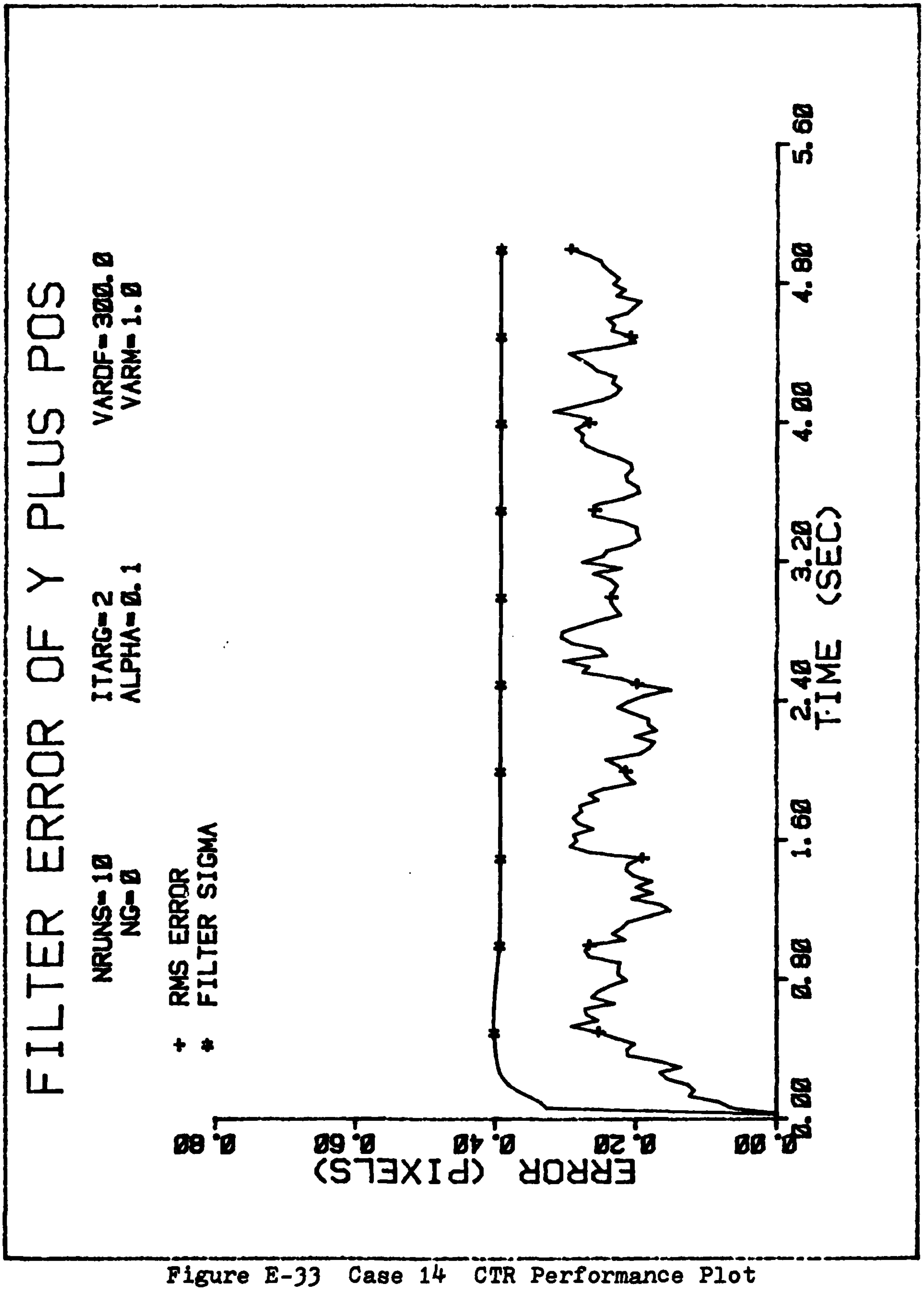


Figure E-33 Case 14 CTR Performance Plot

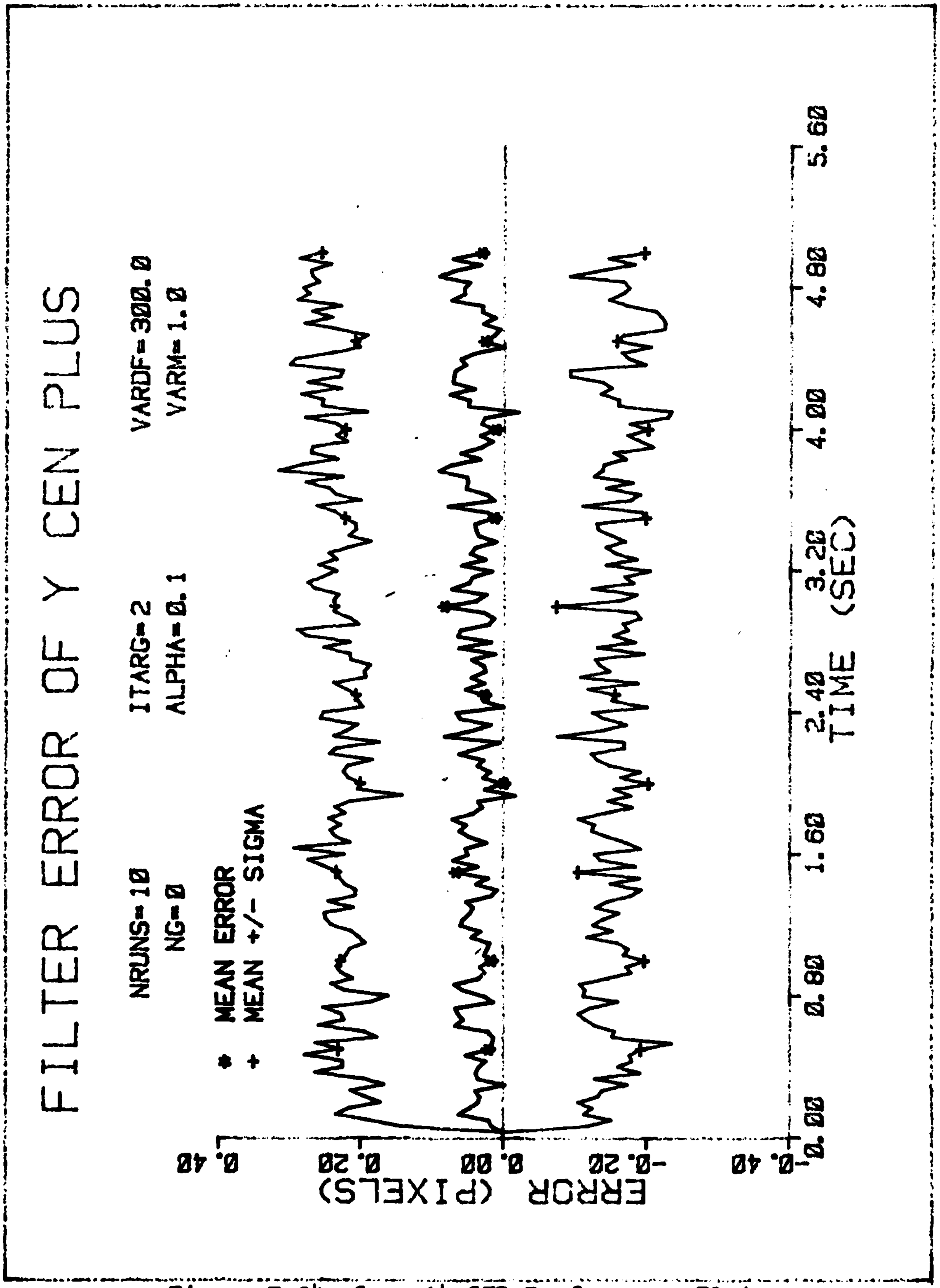


Figure E-34 Case 14 CTR Performance Plot

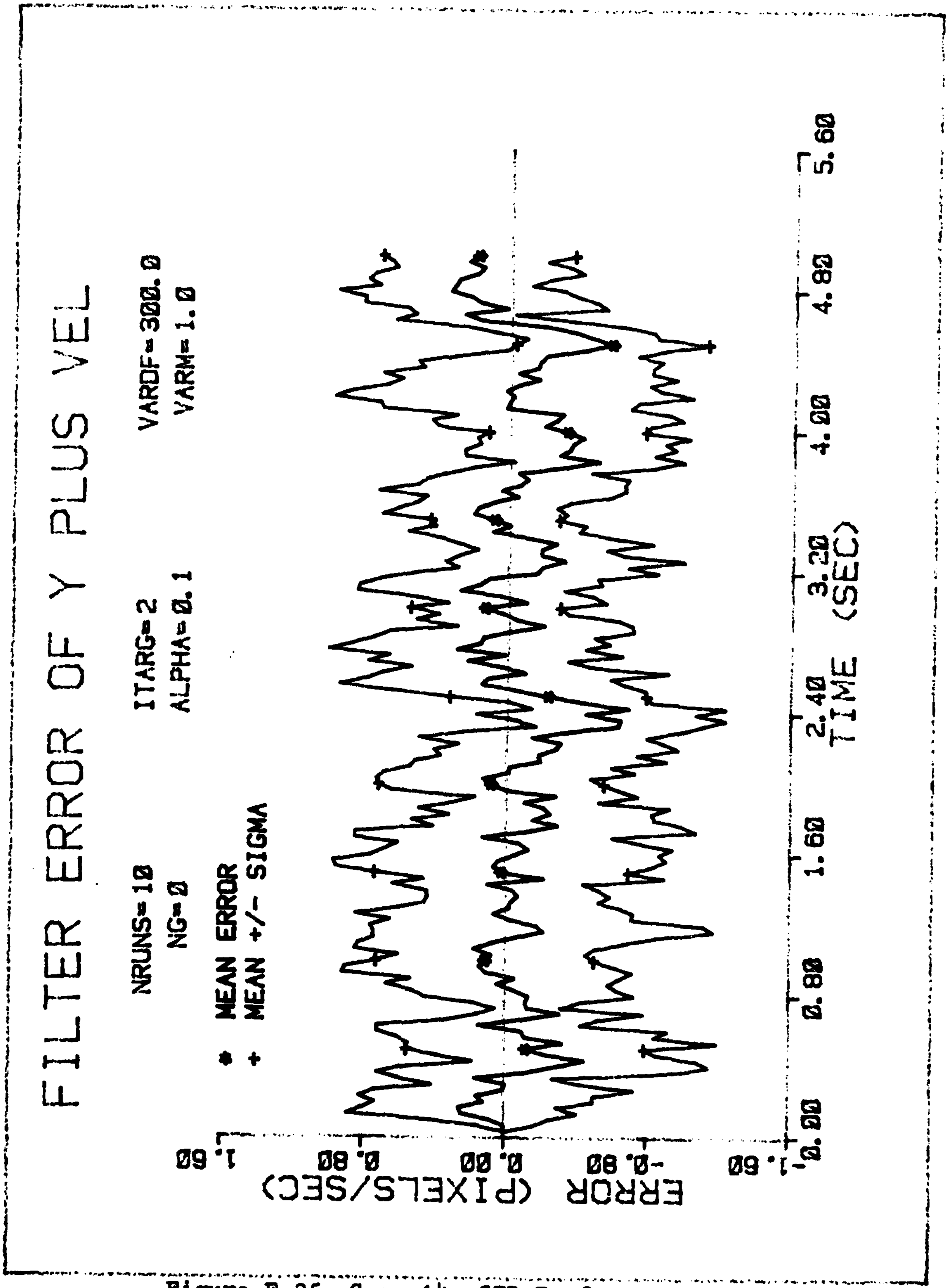


Figure E-35 Case 14 CTR Performance Plot

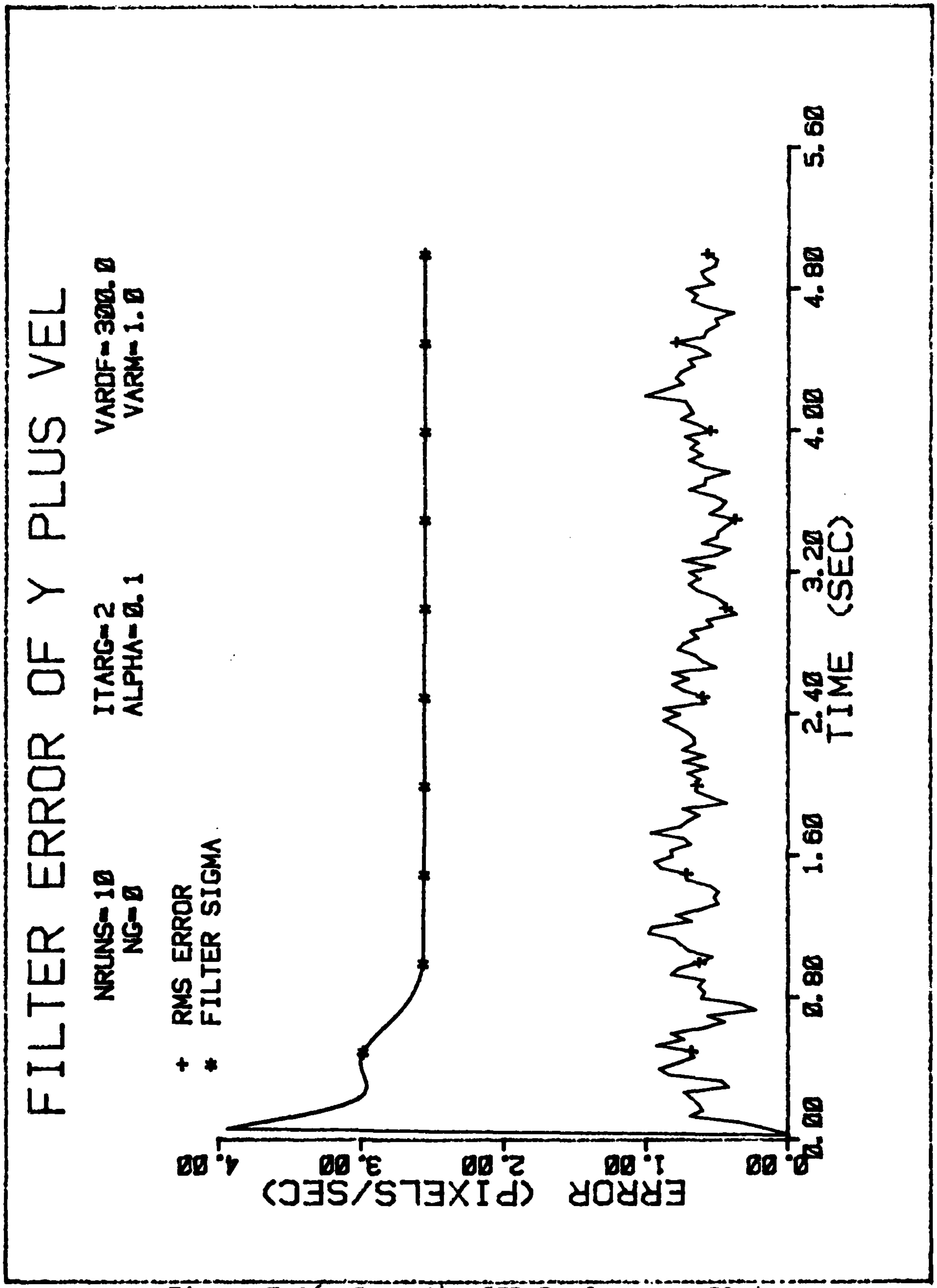


Figure E-36 Case 14 CTR Performance Plot

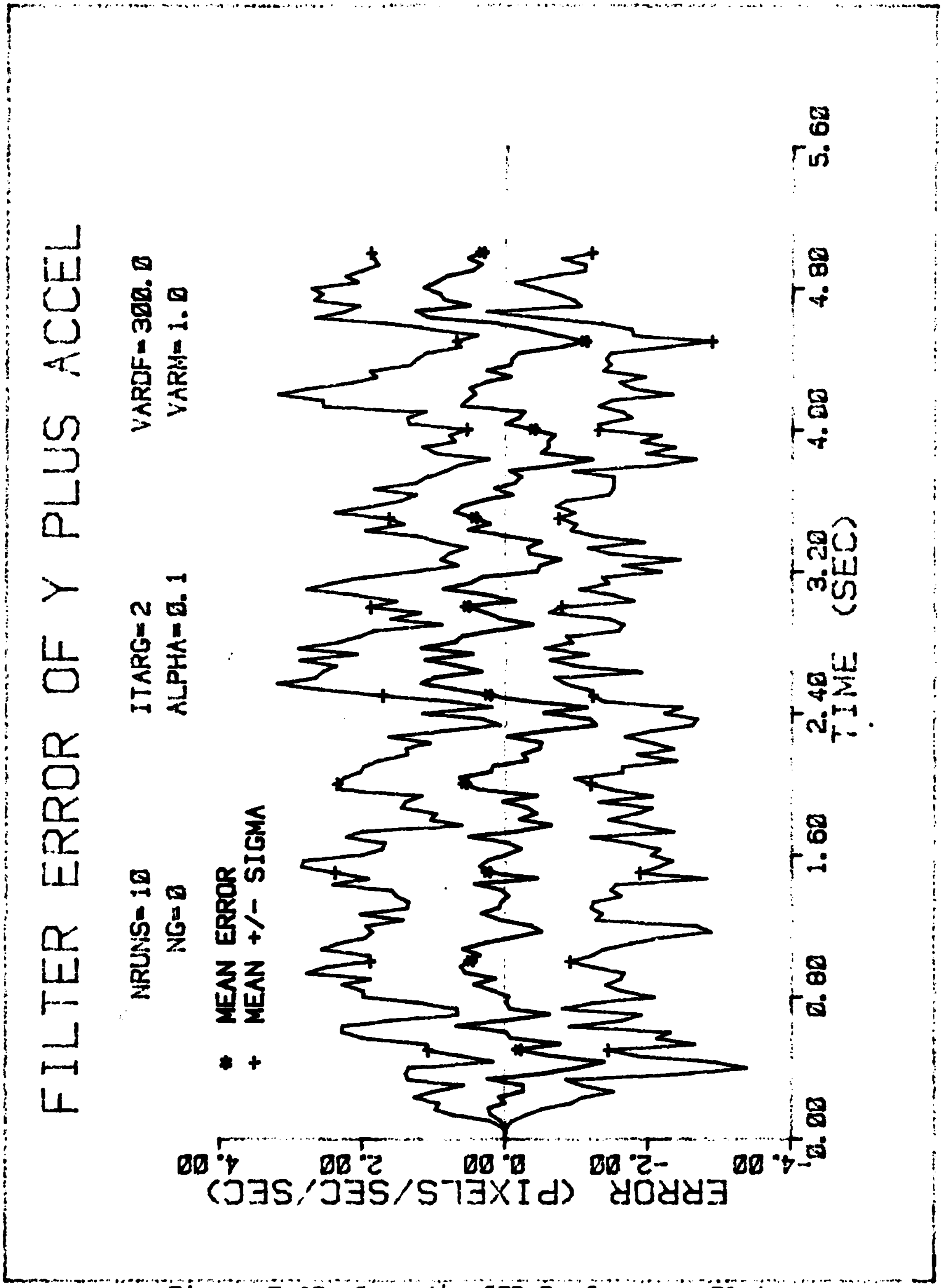


Figure E-37 Case 14 CTR Performance Plot

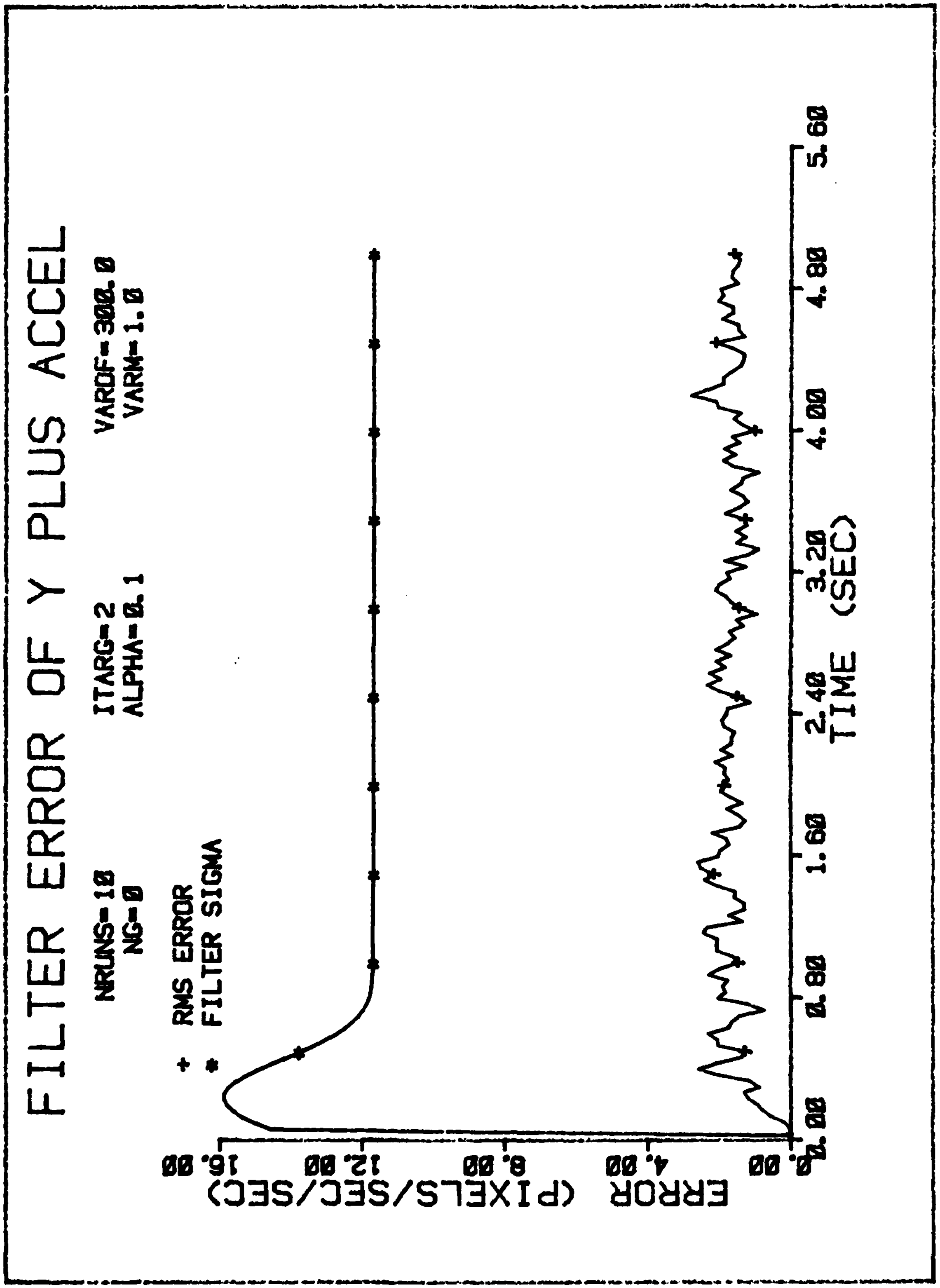


Figure E-38 Case 14 CTR Performance Plot

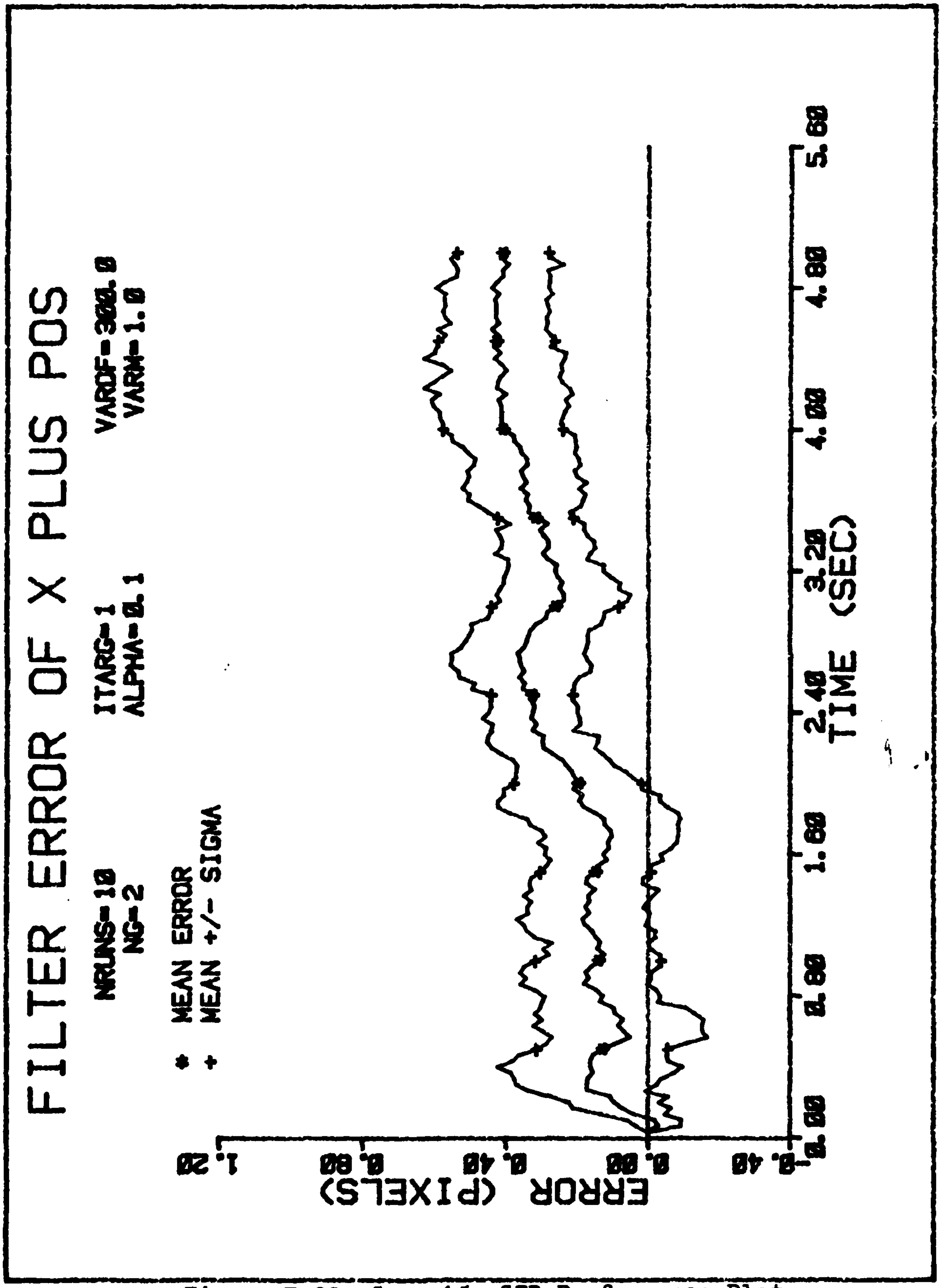


Figure E-39 Case 15 CTR Performance Plot

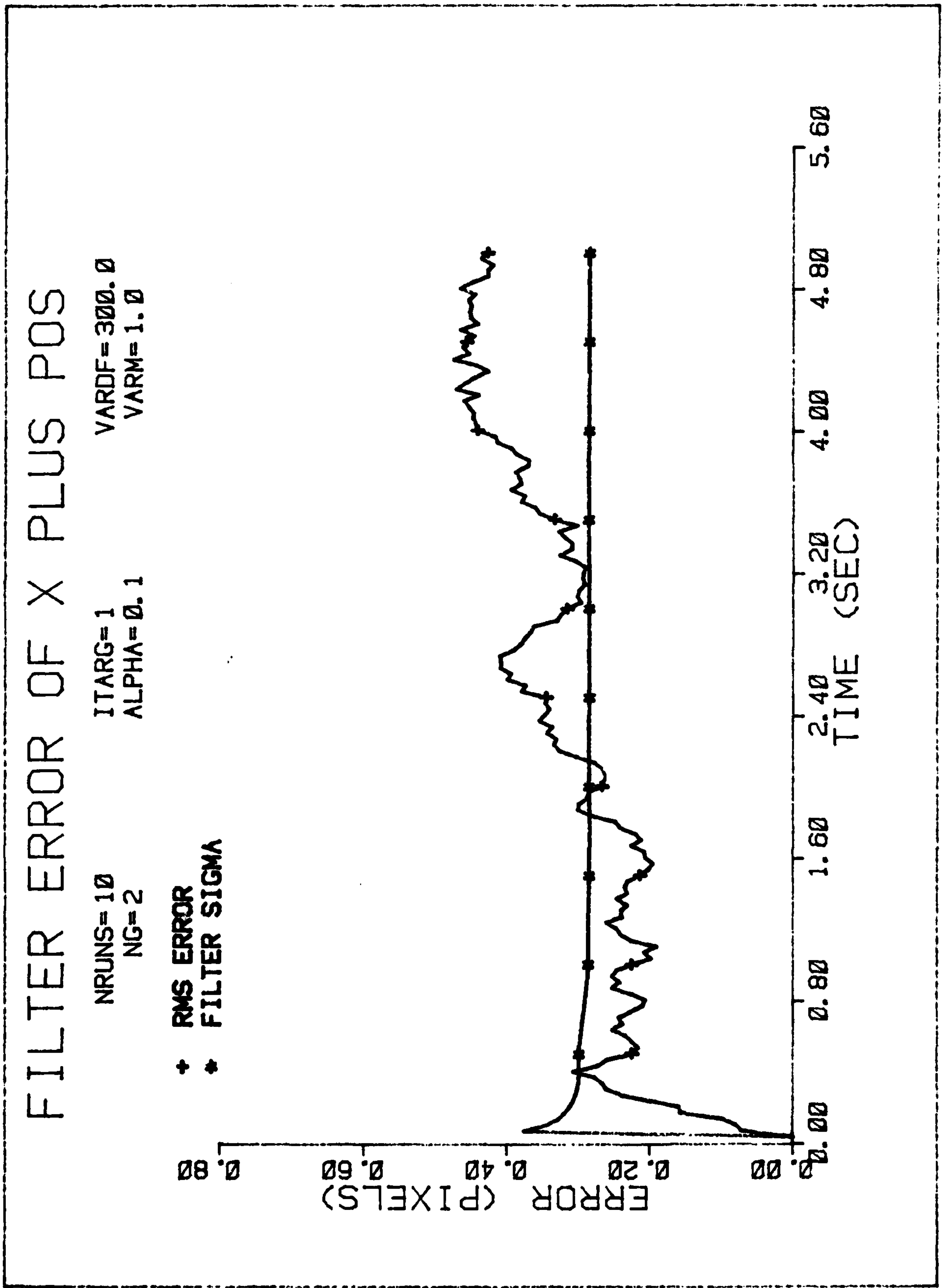


Figure E-40 Case 15 CTR Performance Plot

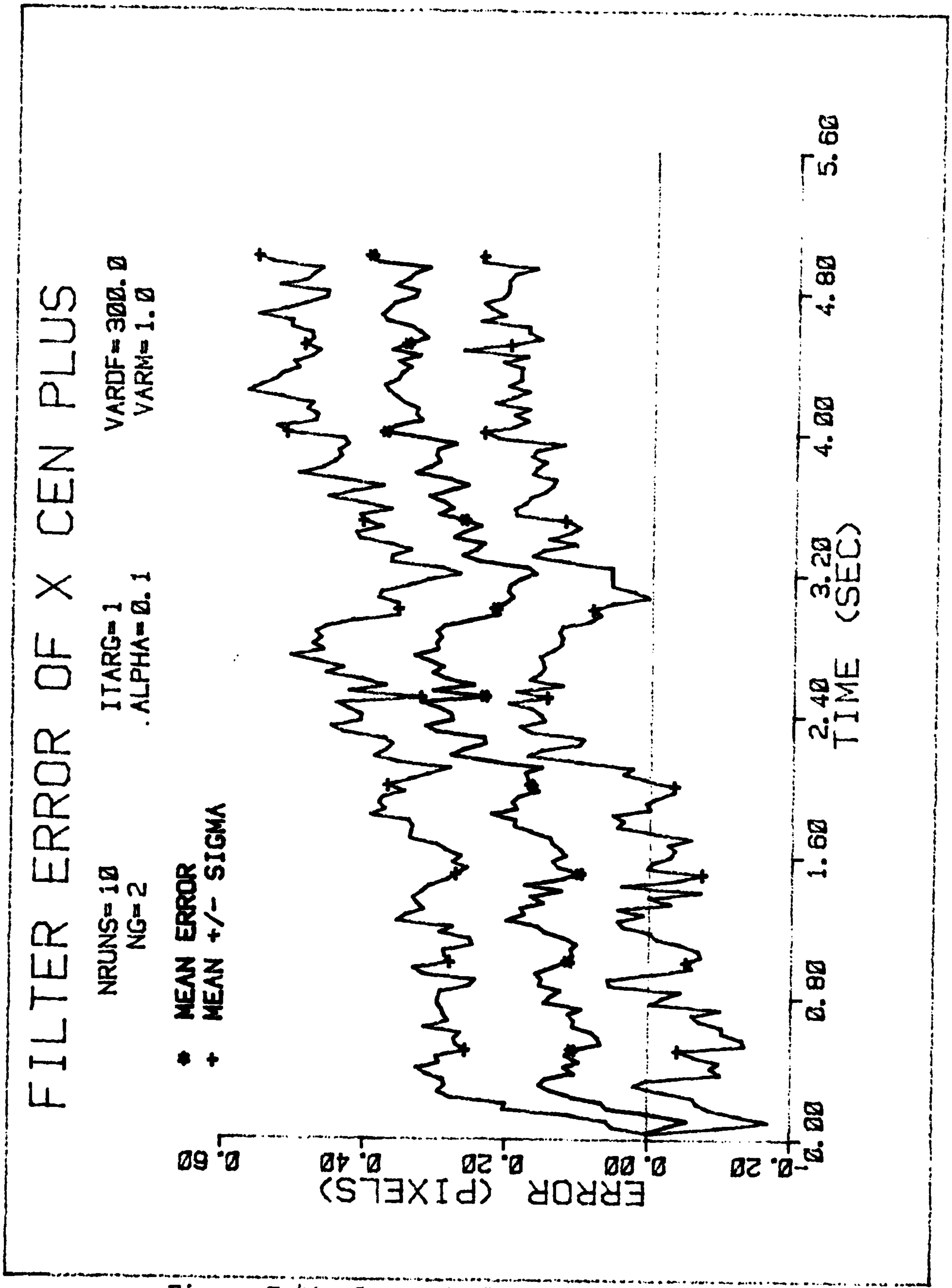


Figure E-41 Case 15 CTR Performance Plot

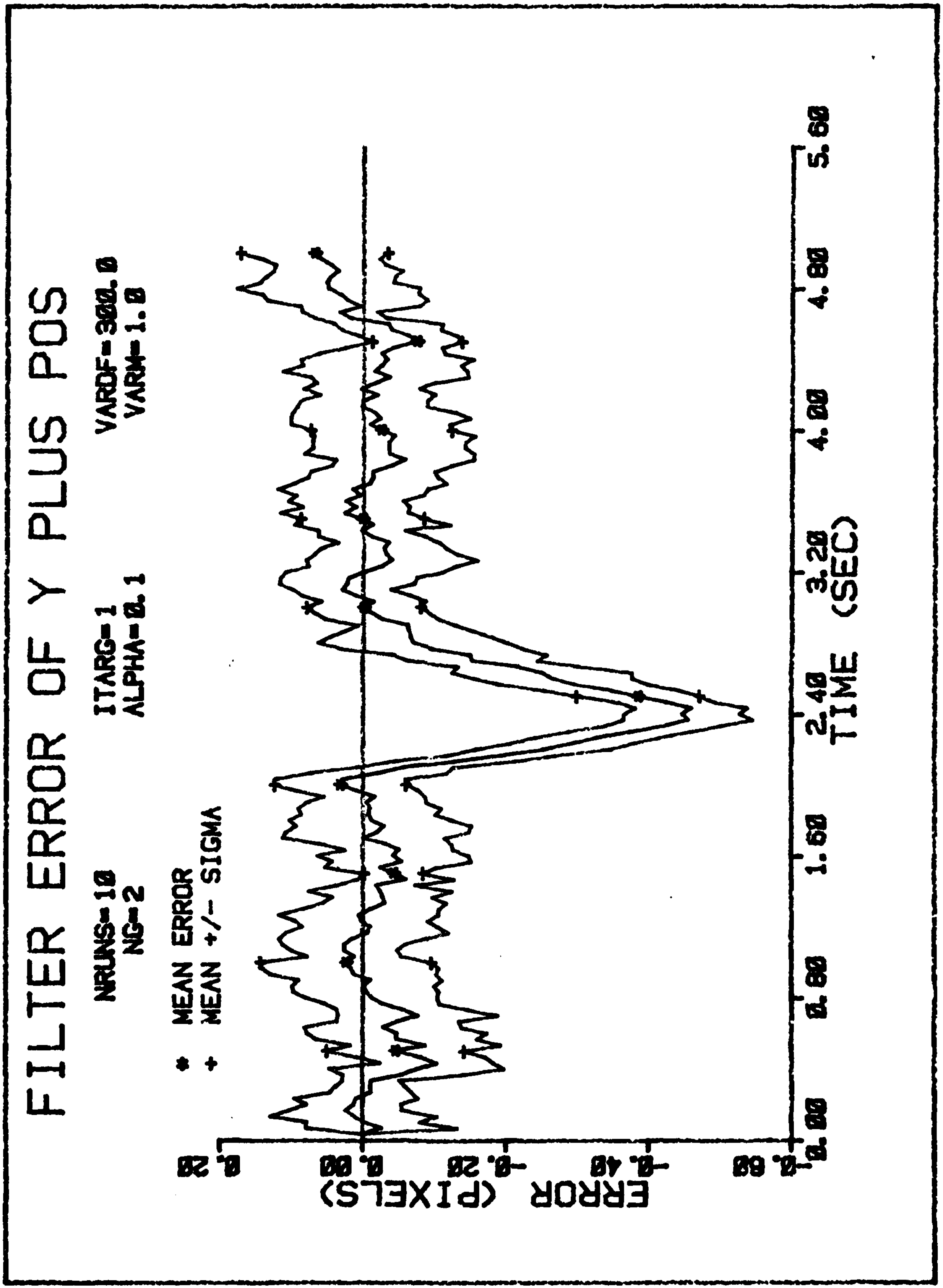


Figure E-42 Case 15 CTR Performance Plot

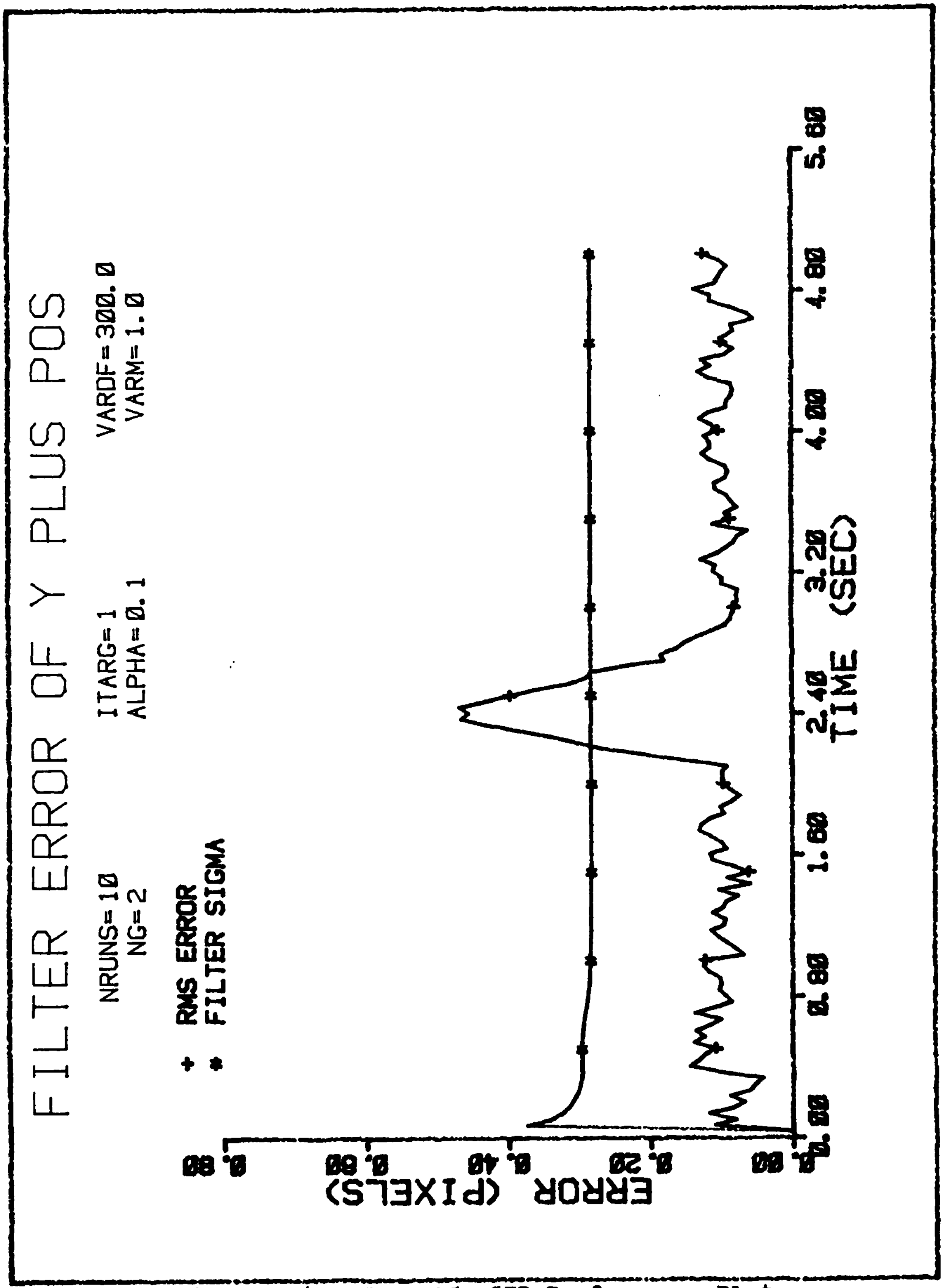


Figure E-43 Case 15 CTR Performance Plot

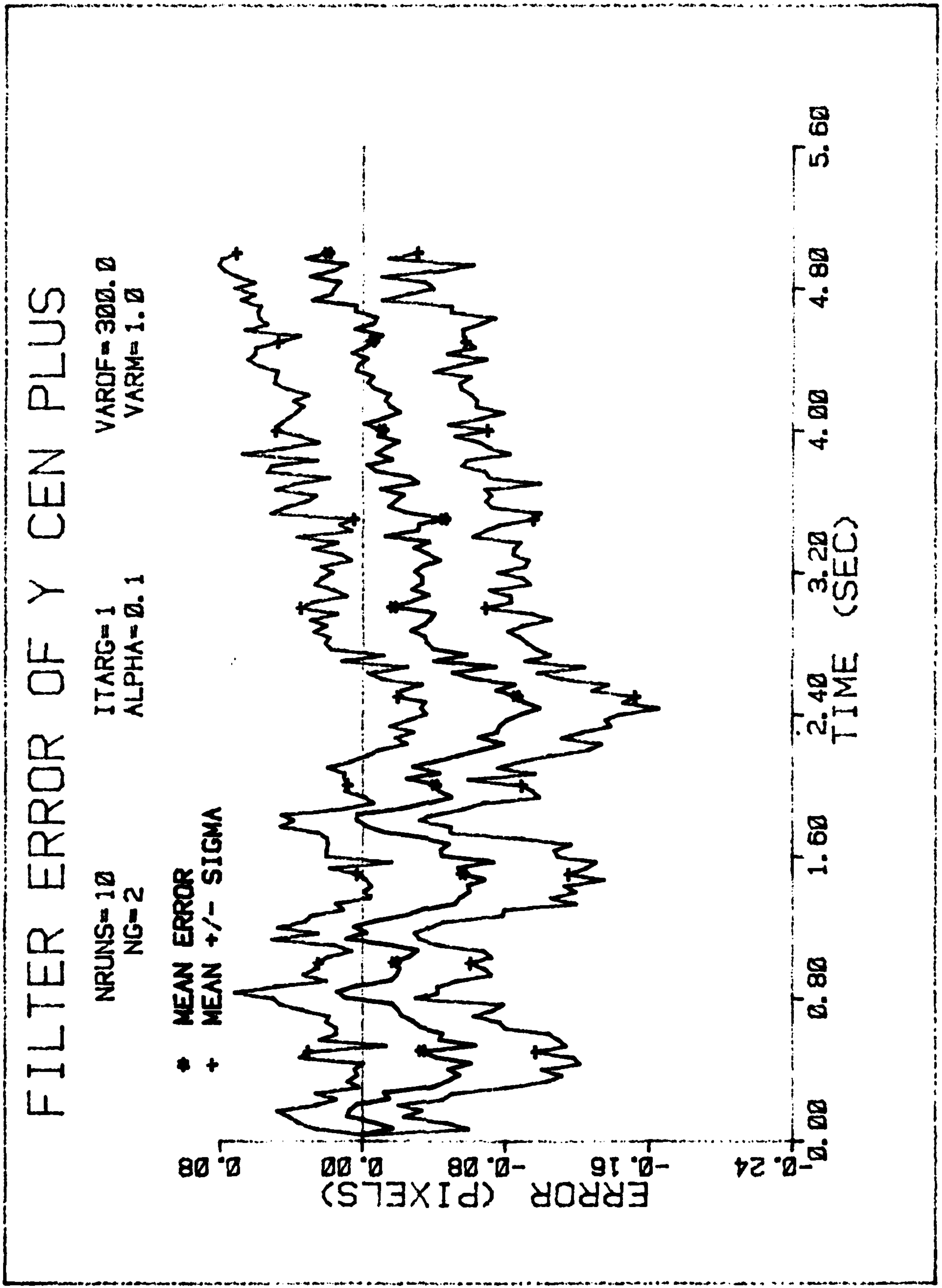


Figure E-44 Case 15 CTR Performance Plot

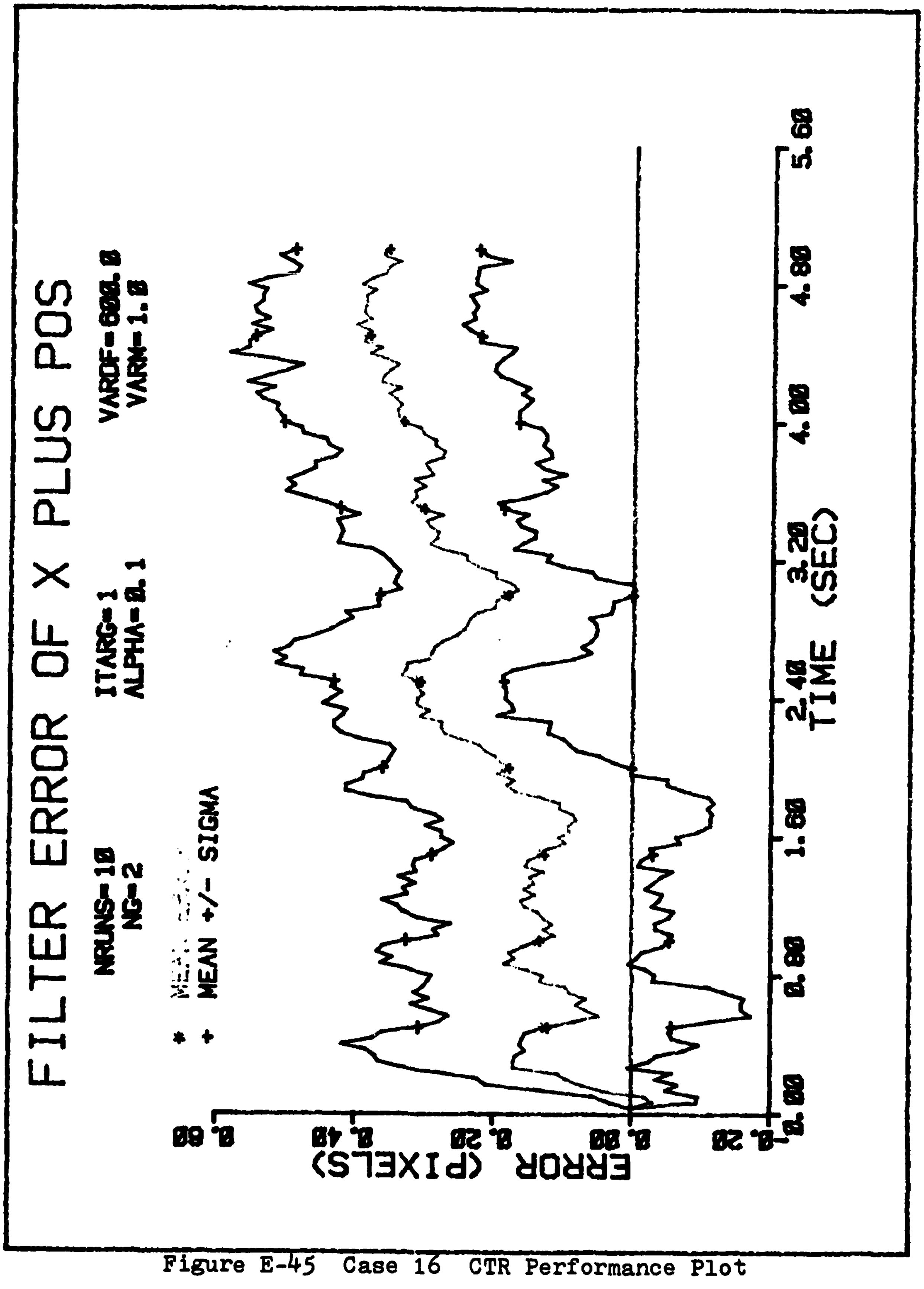


Figure E-45 Case 16 CTR Performance Plot

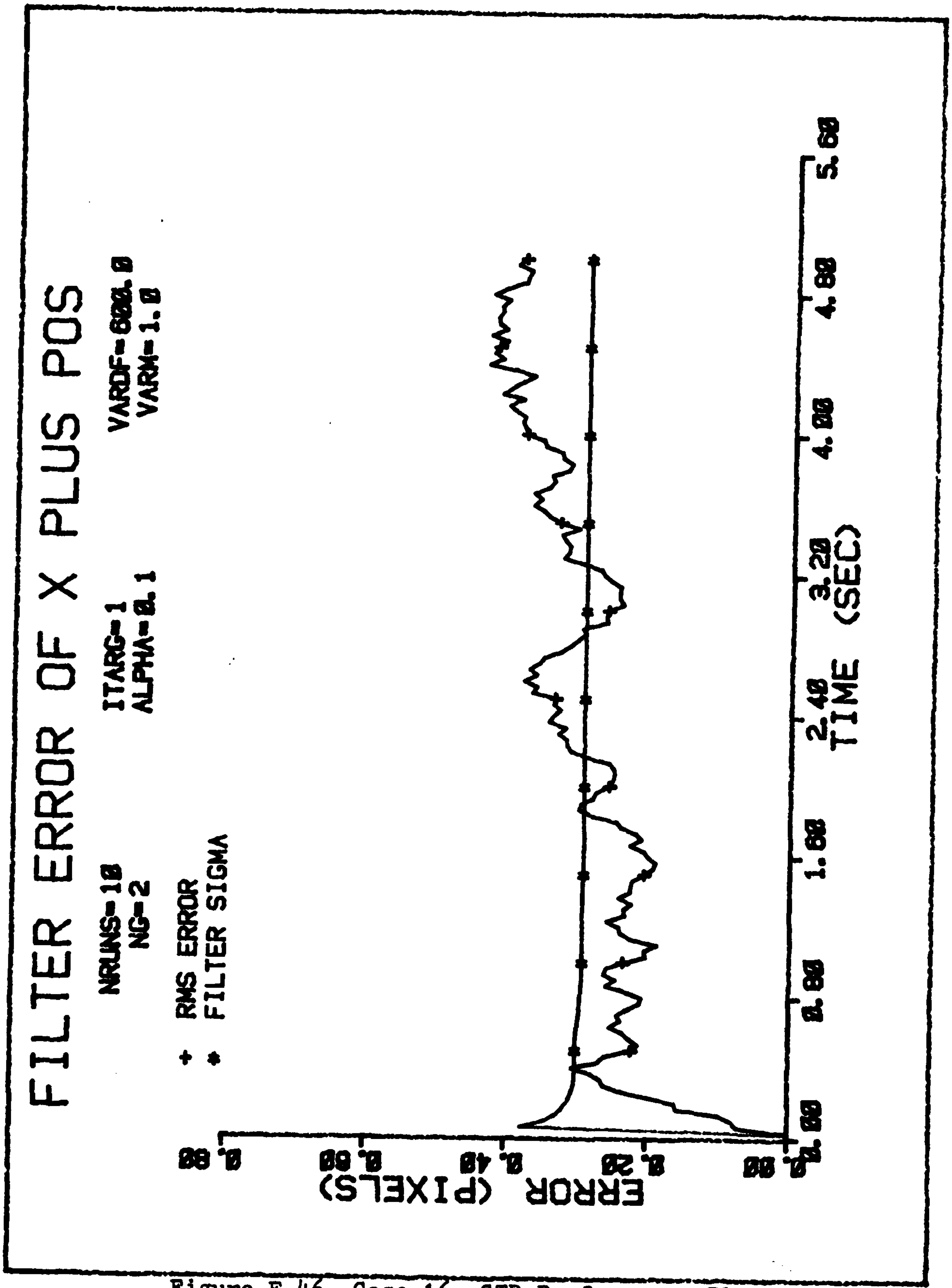


Figure E-46 Case 16 CTR Performance Plot

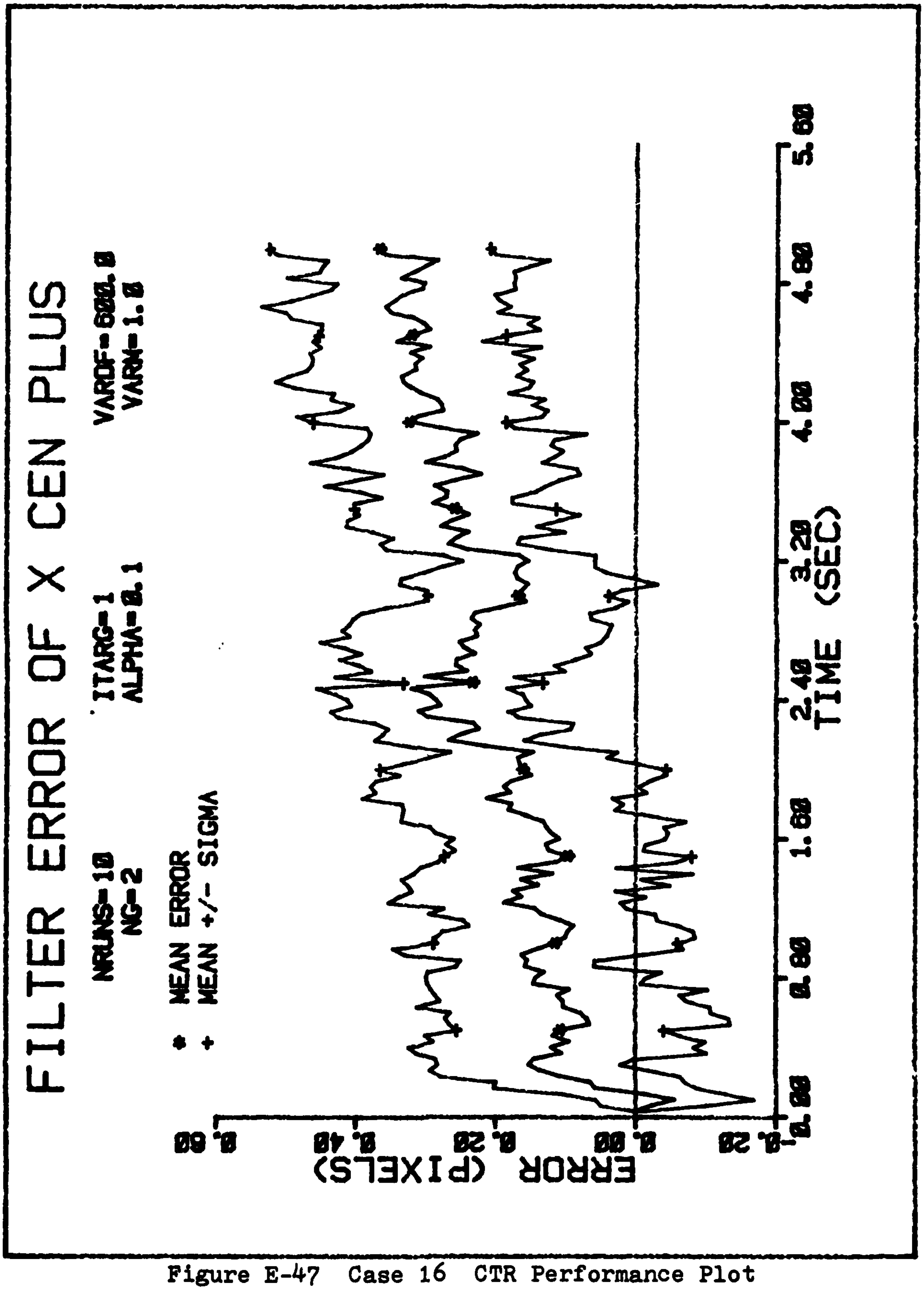


Figure E-47 Case 16 CTR Performance Plot

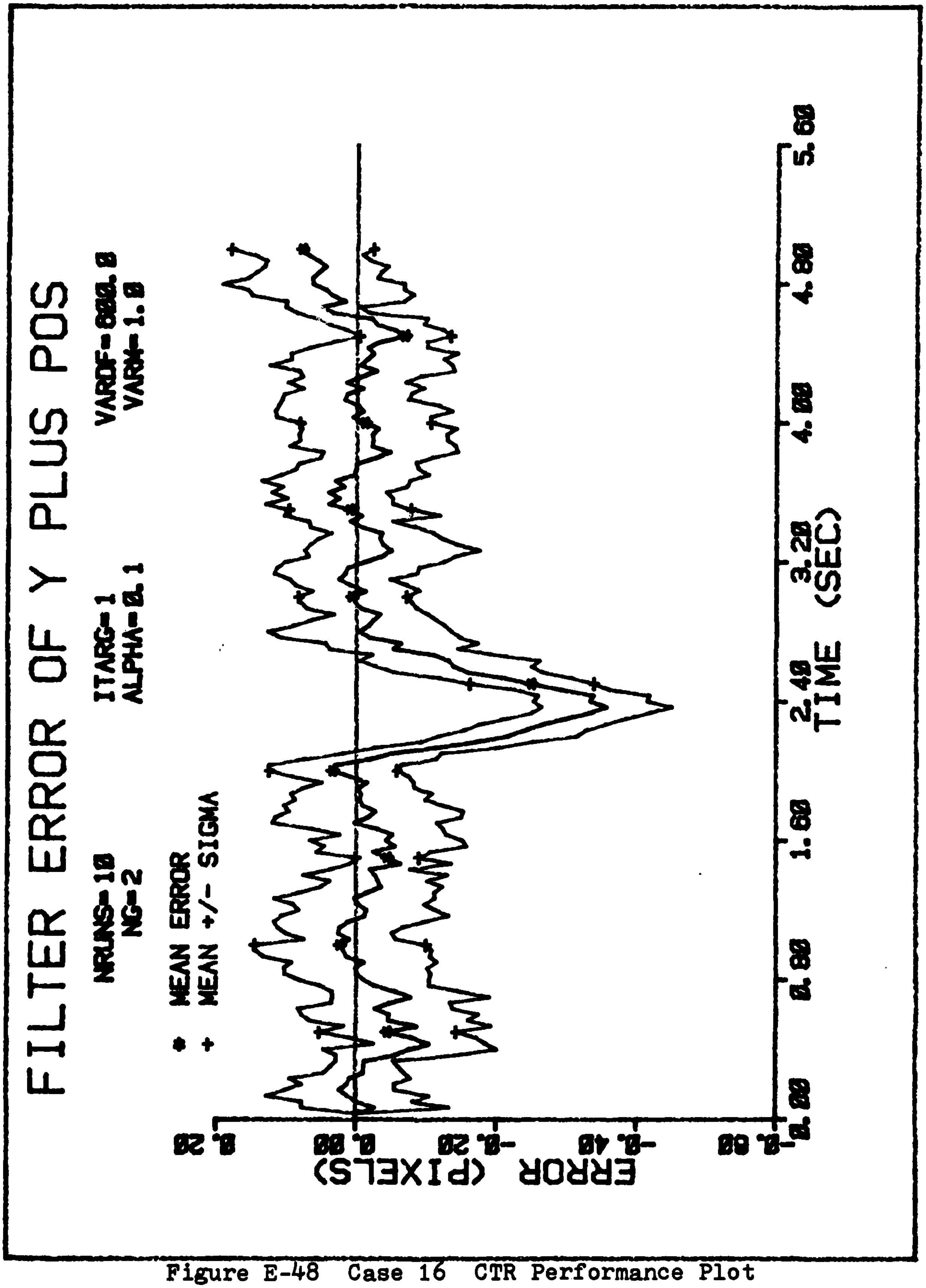


Figure E-48 Case 16 CTR Performance Plot

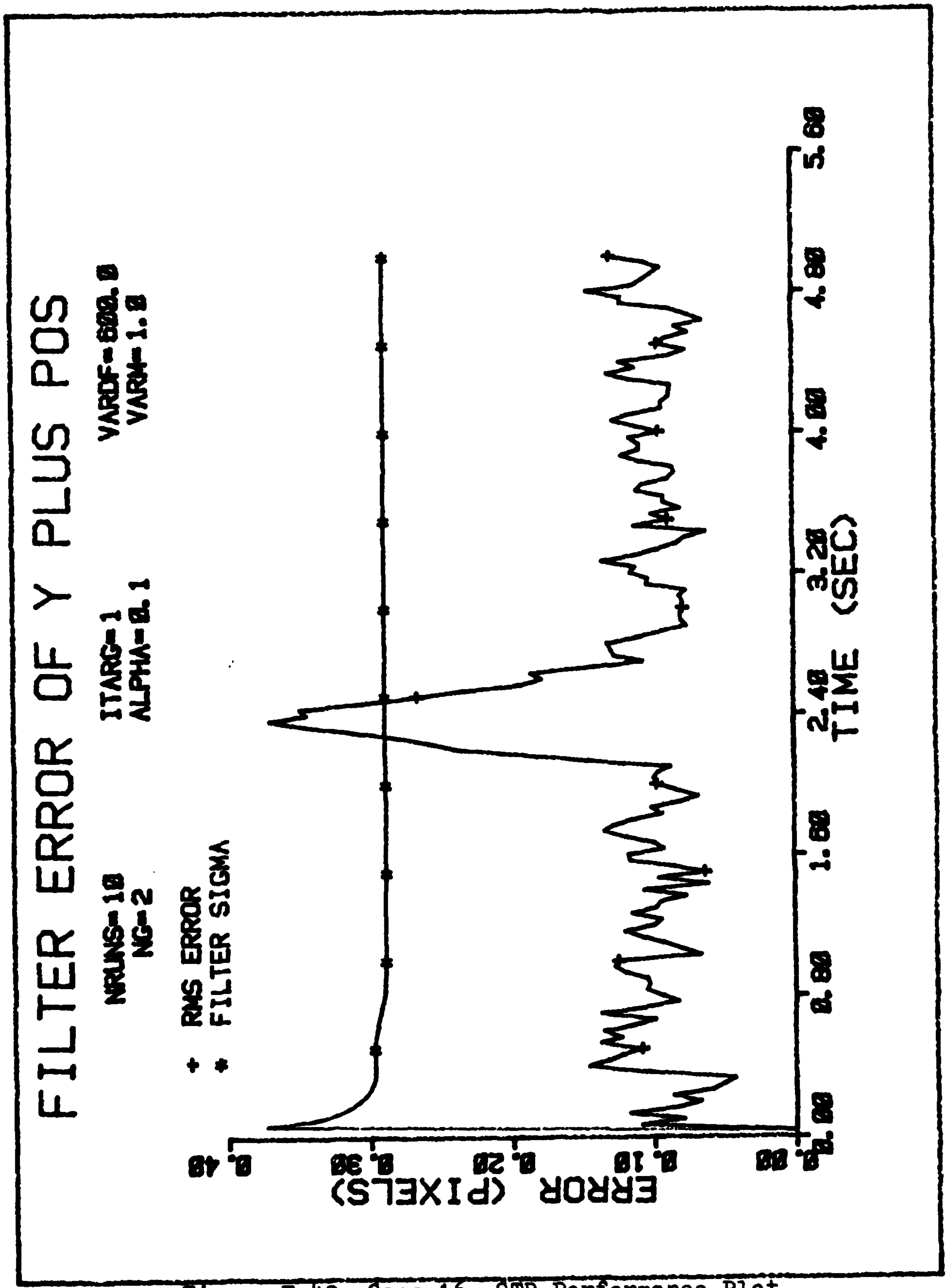


Figure E-49 Case 16 CTR Performance Plot

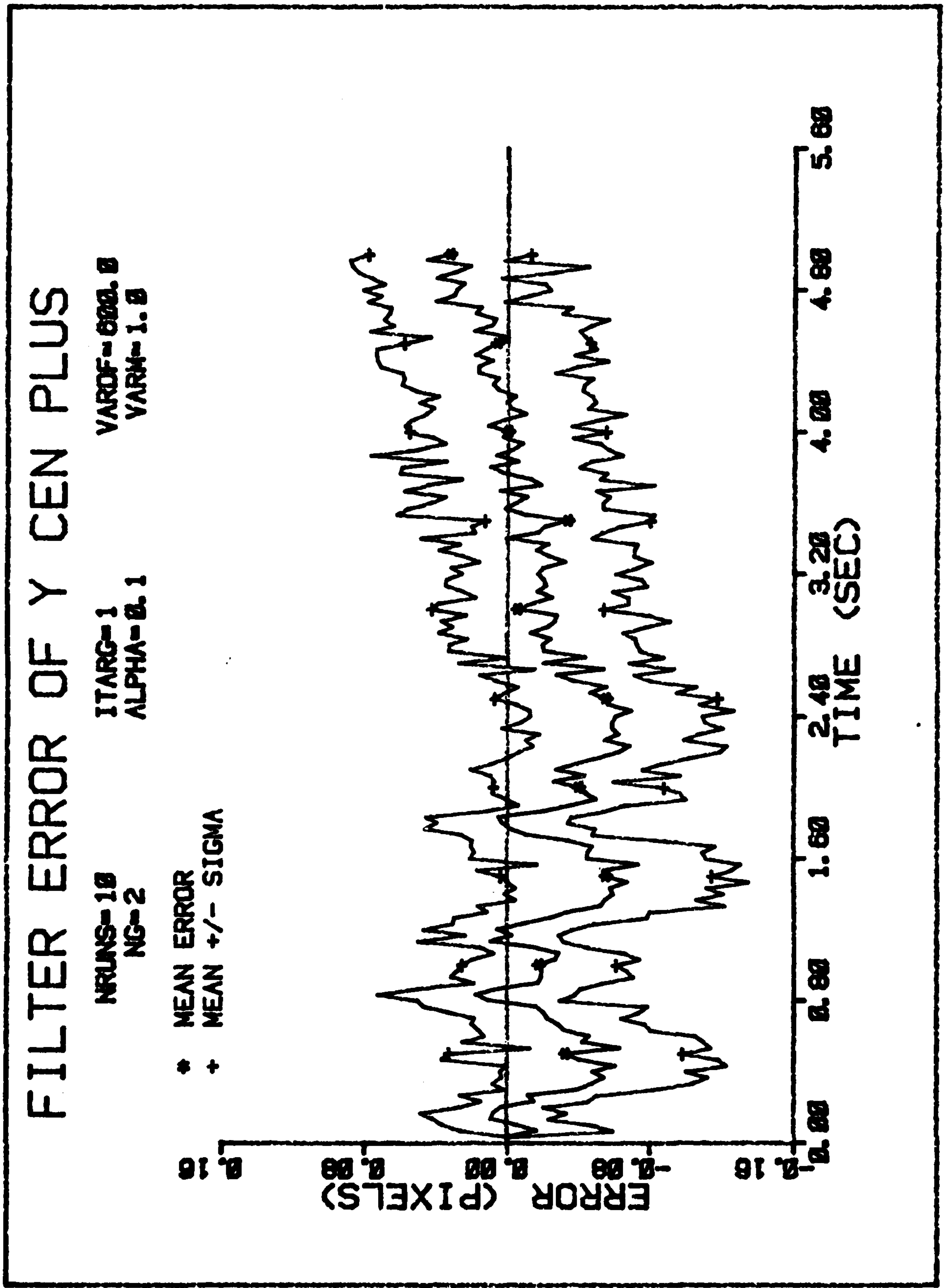


Figure E-50 Case 16 CTR Performance Plot

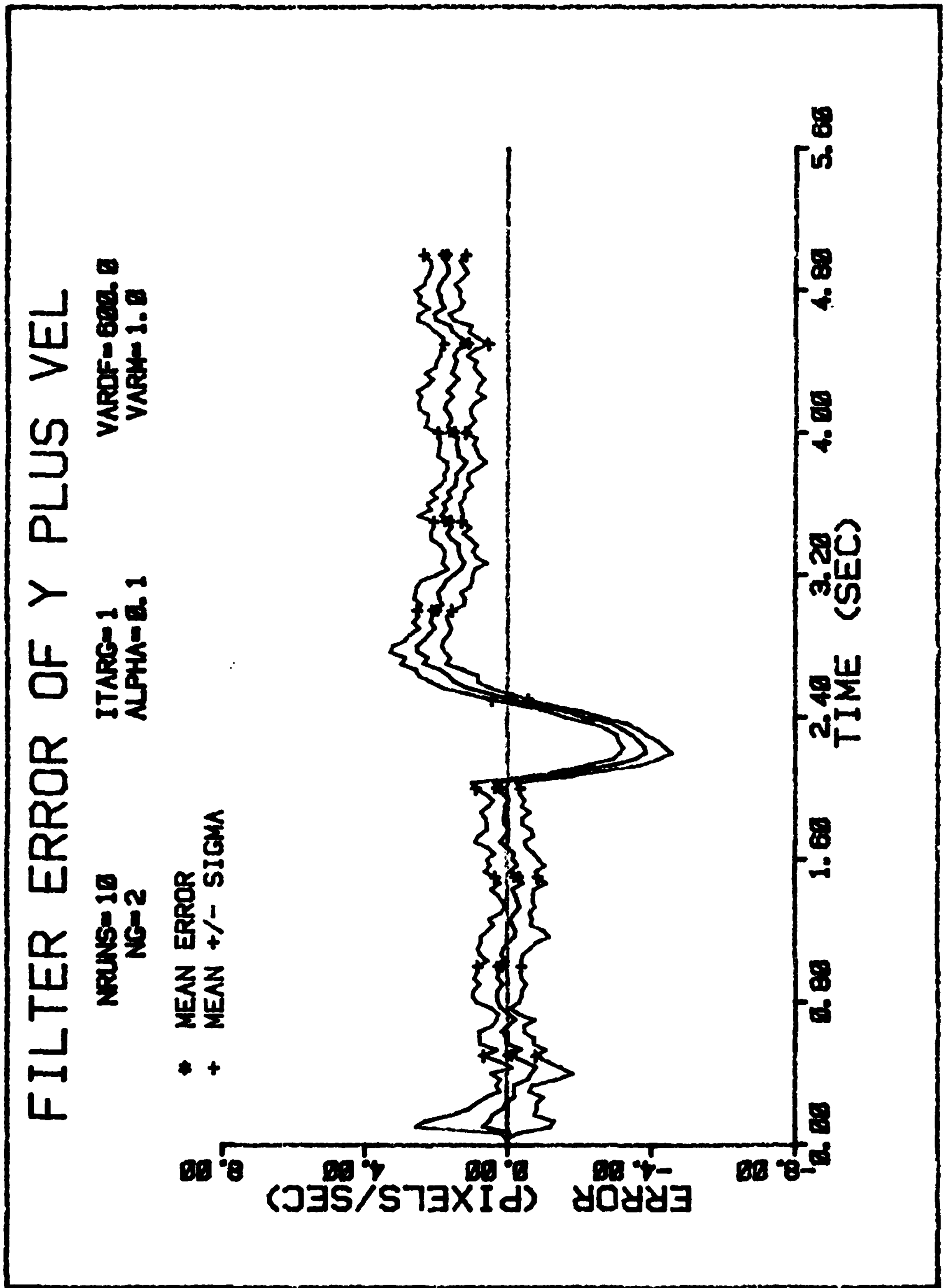


Figure E-51 Case 16 CTR Performance Plot

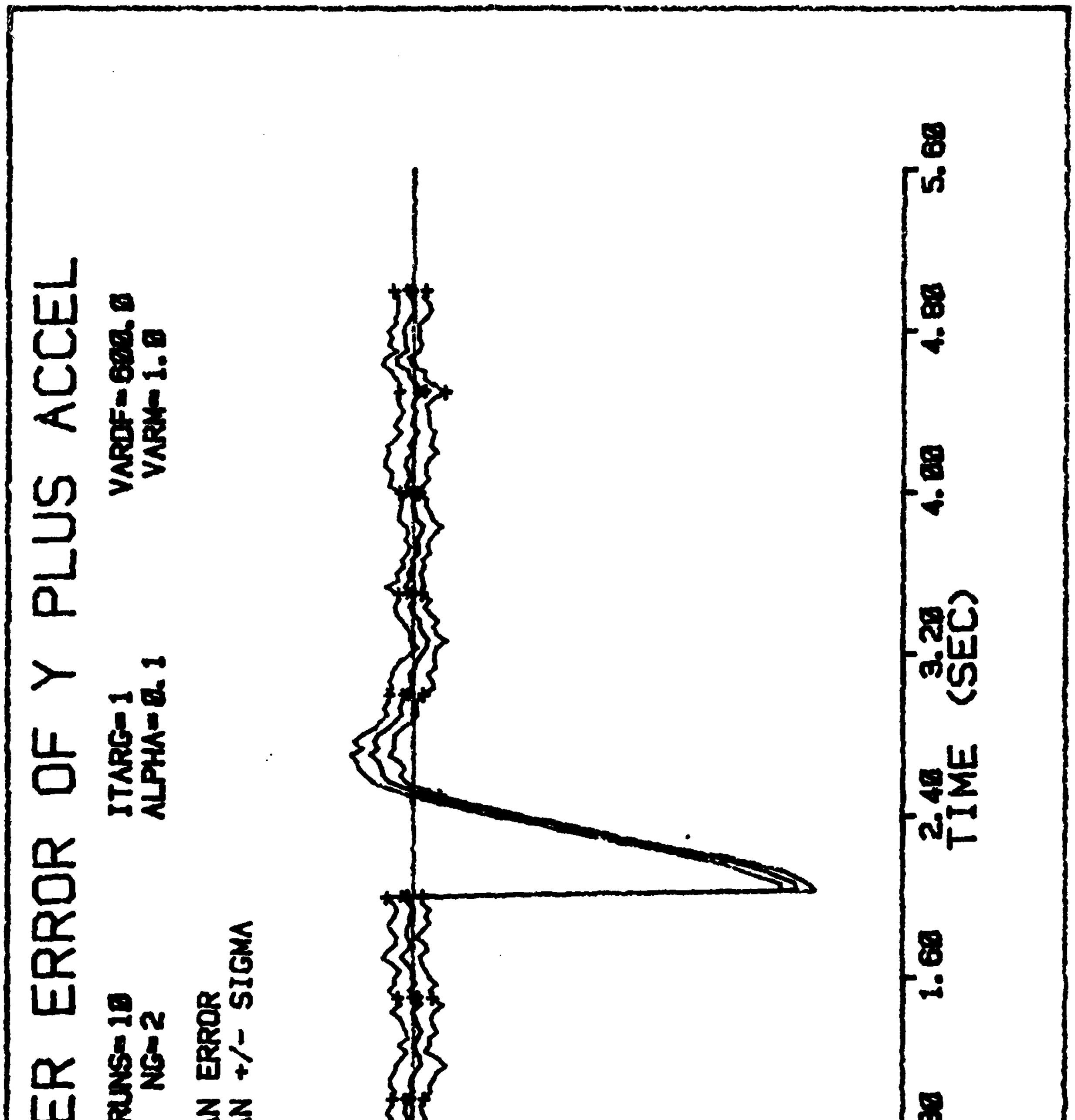


Figure E-52 Case 16 CTR Performance Plot

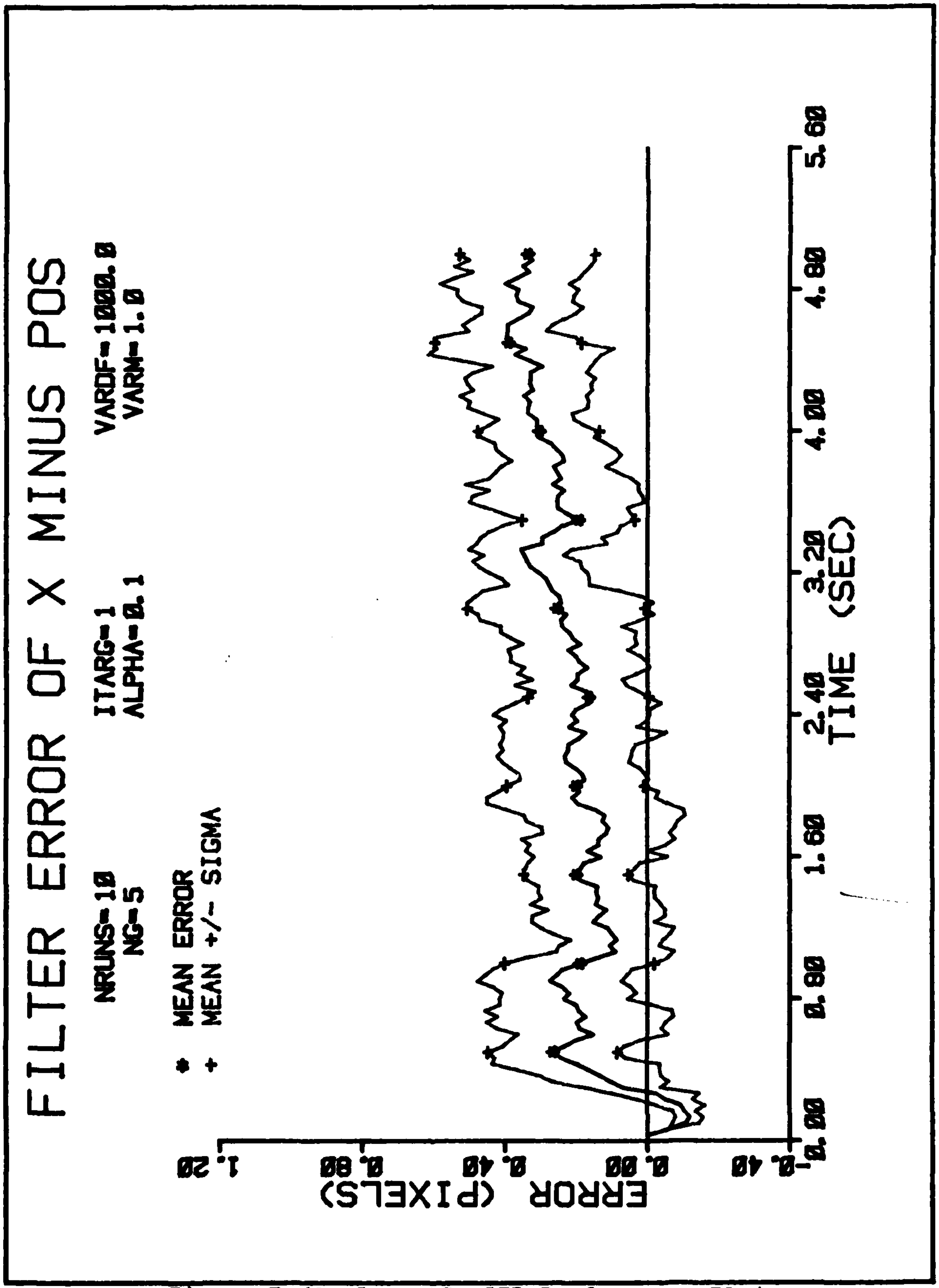


Figure E-53 Case 17 CTR Performance Plot

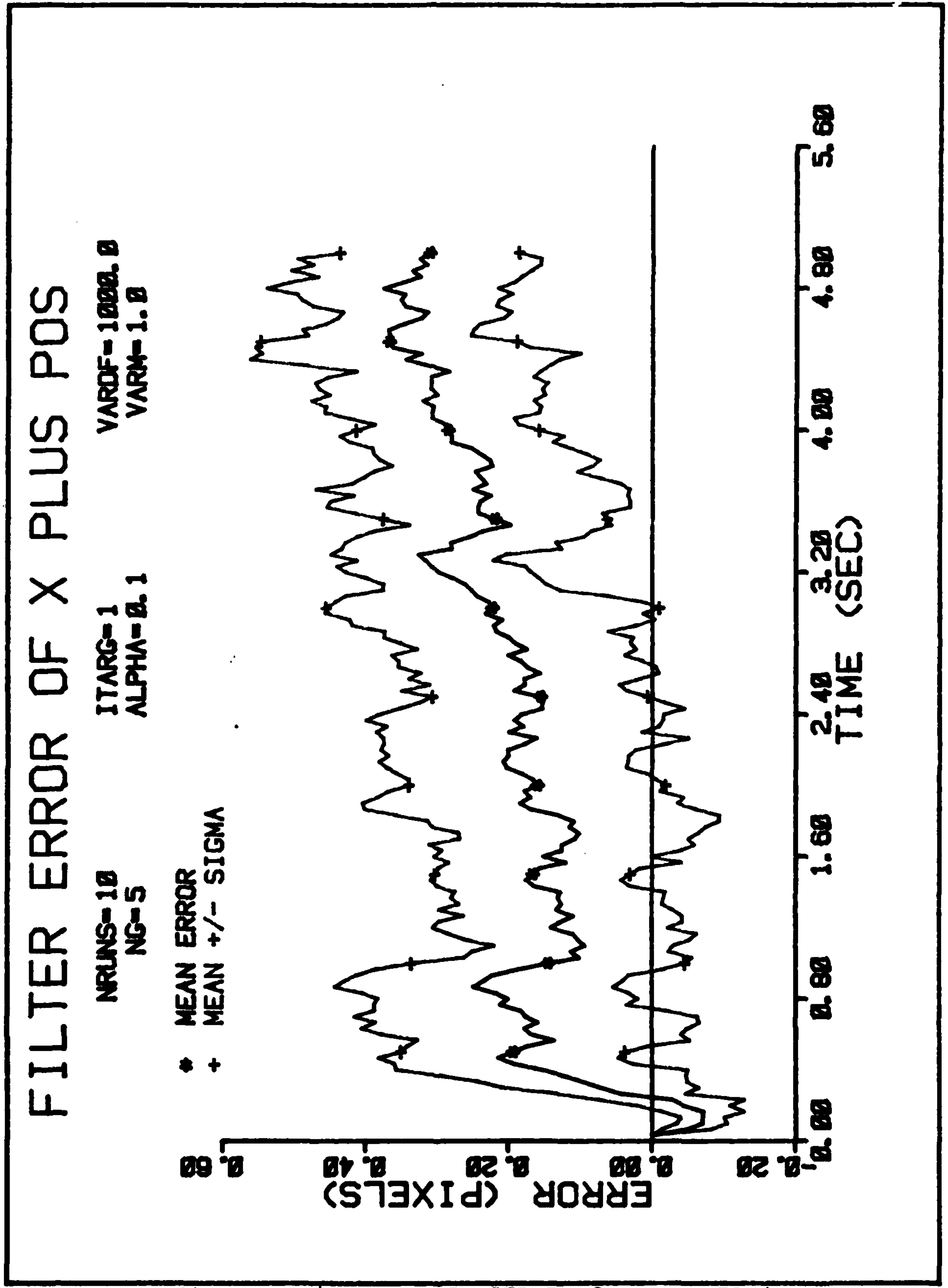


Figure E-54 Case 17 CTR Performance Plot

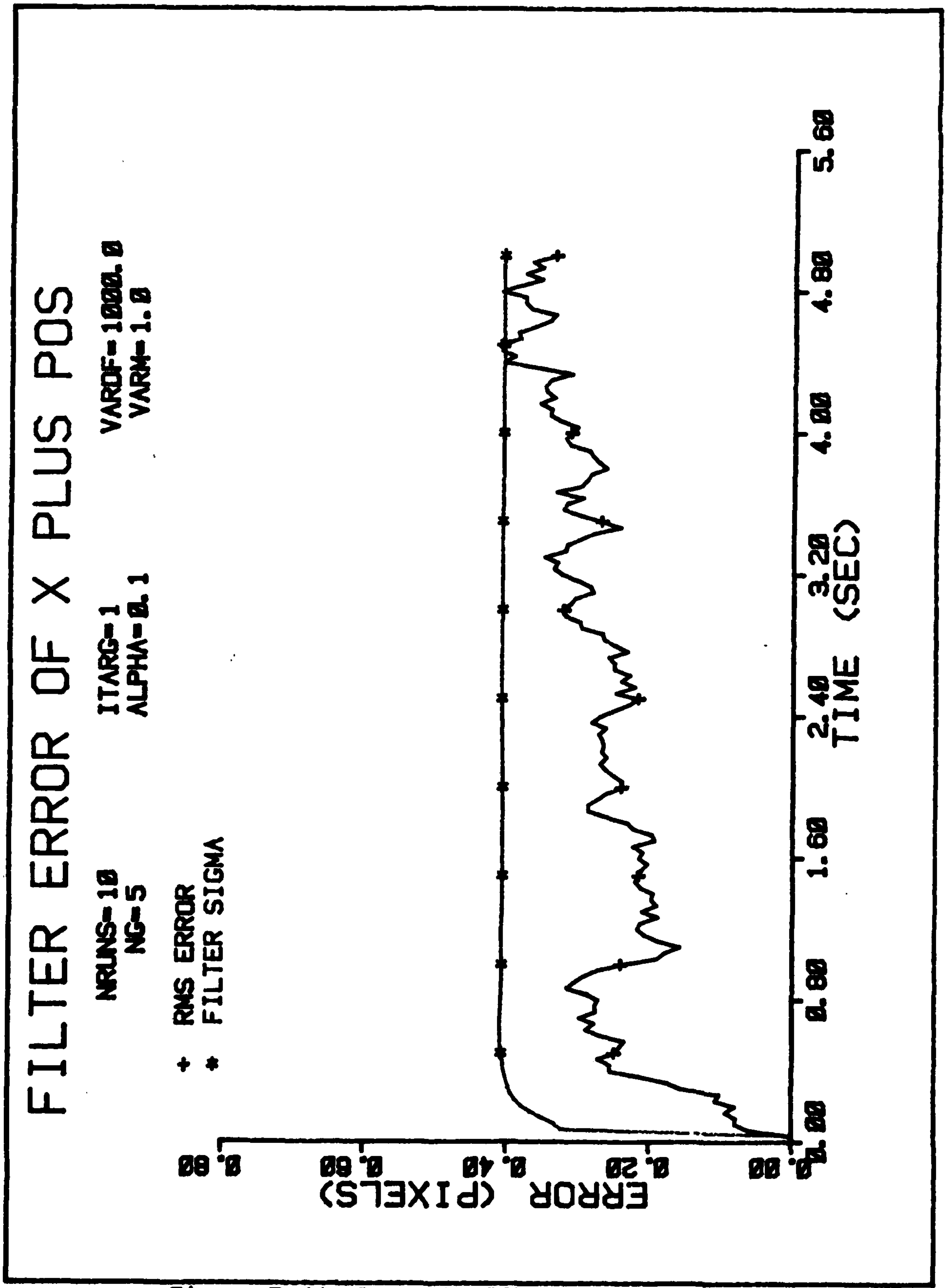


Figure E-55 Case 17 CTR Performance Plot

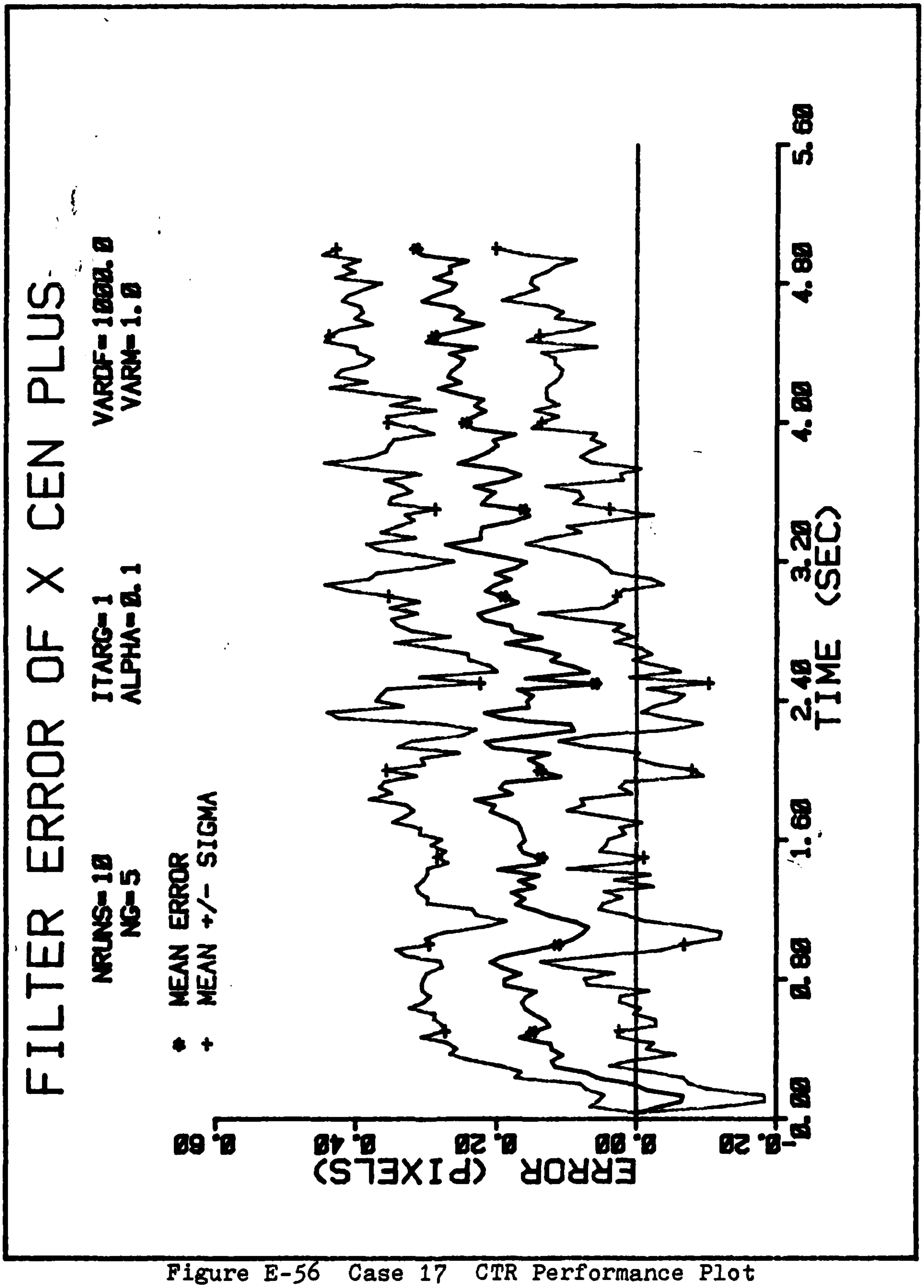


Figure E-56 Case 17 CTR Performance Plot

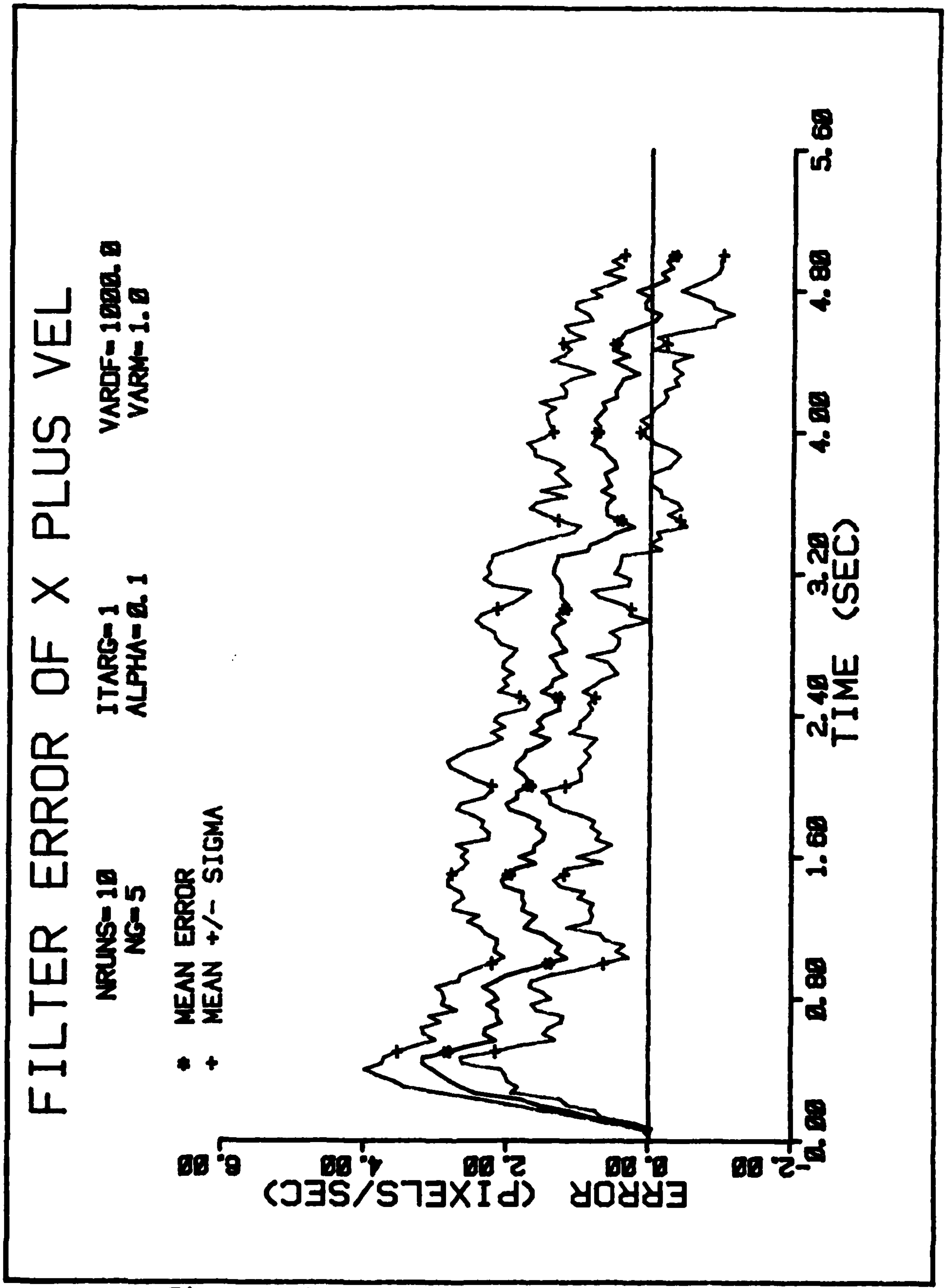


Figure E-57 Case 17 CTR Performance Plot

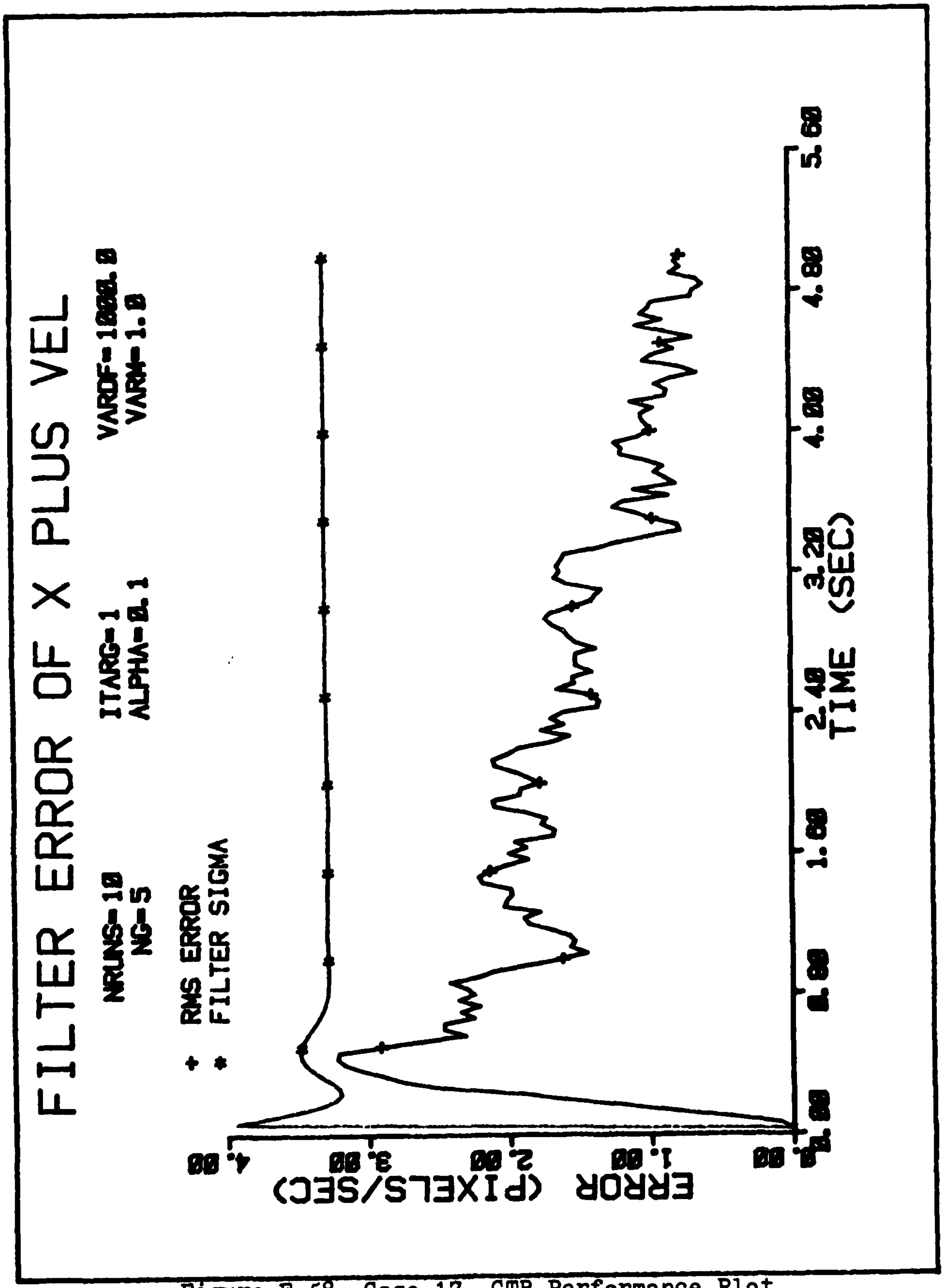


Figure E-58 Case 17 CTR Performance Plot

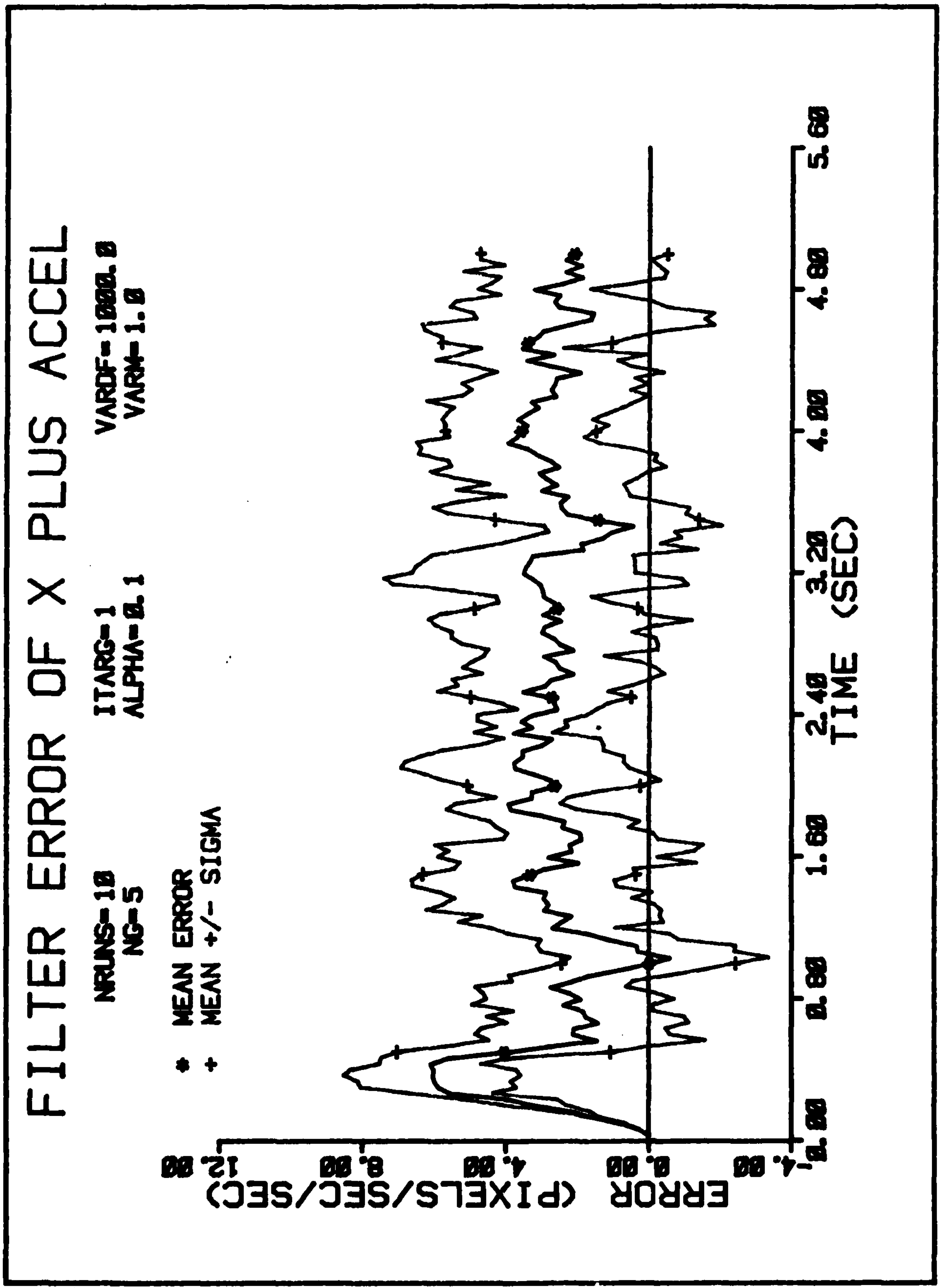


Figure E-59 Case 17 CTR Performance Plot

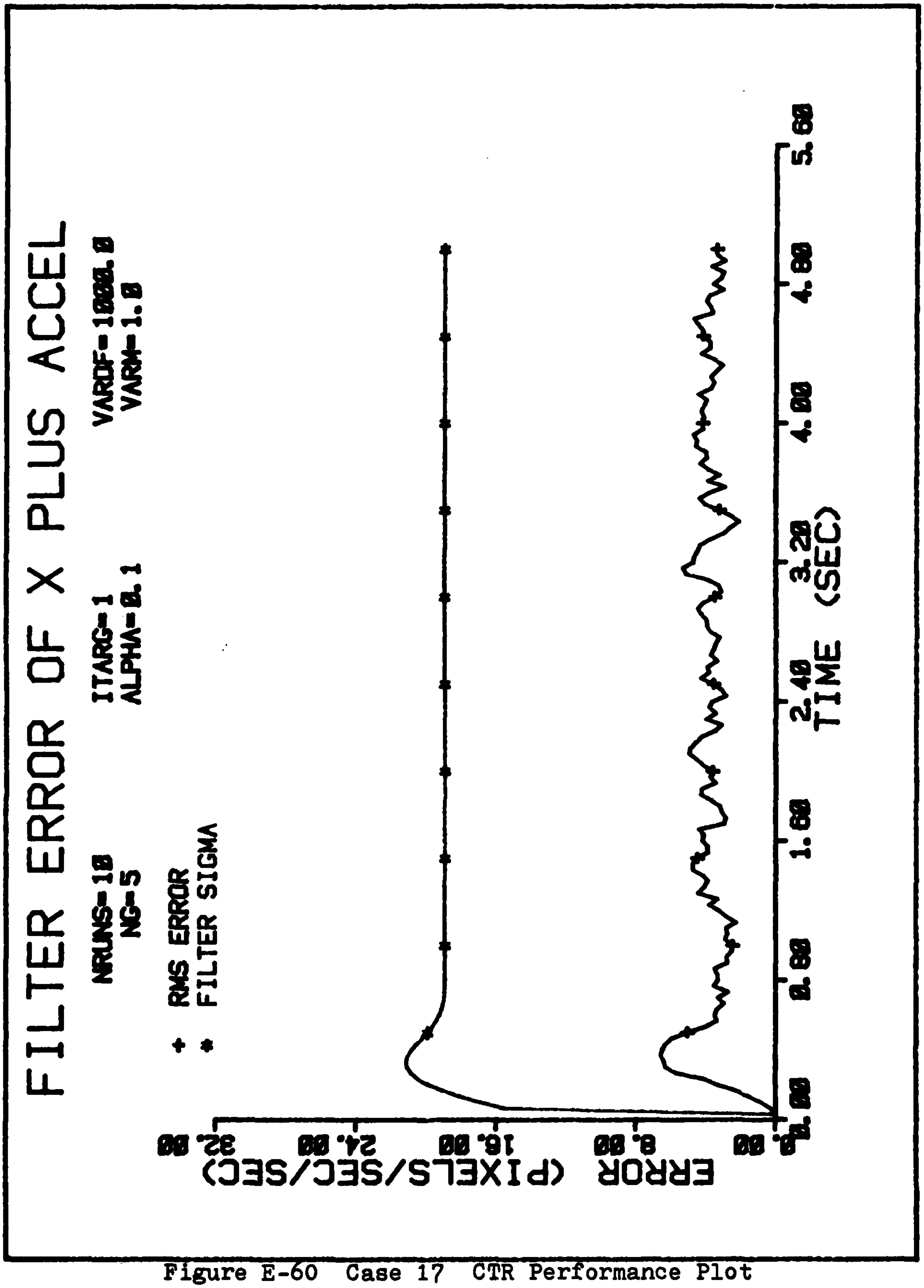


Figure E-60 Case 17 CTR Performance Plot

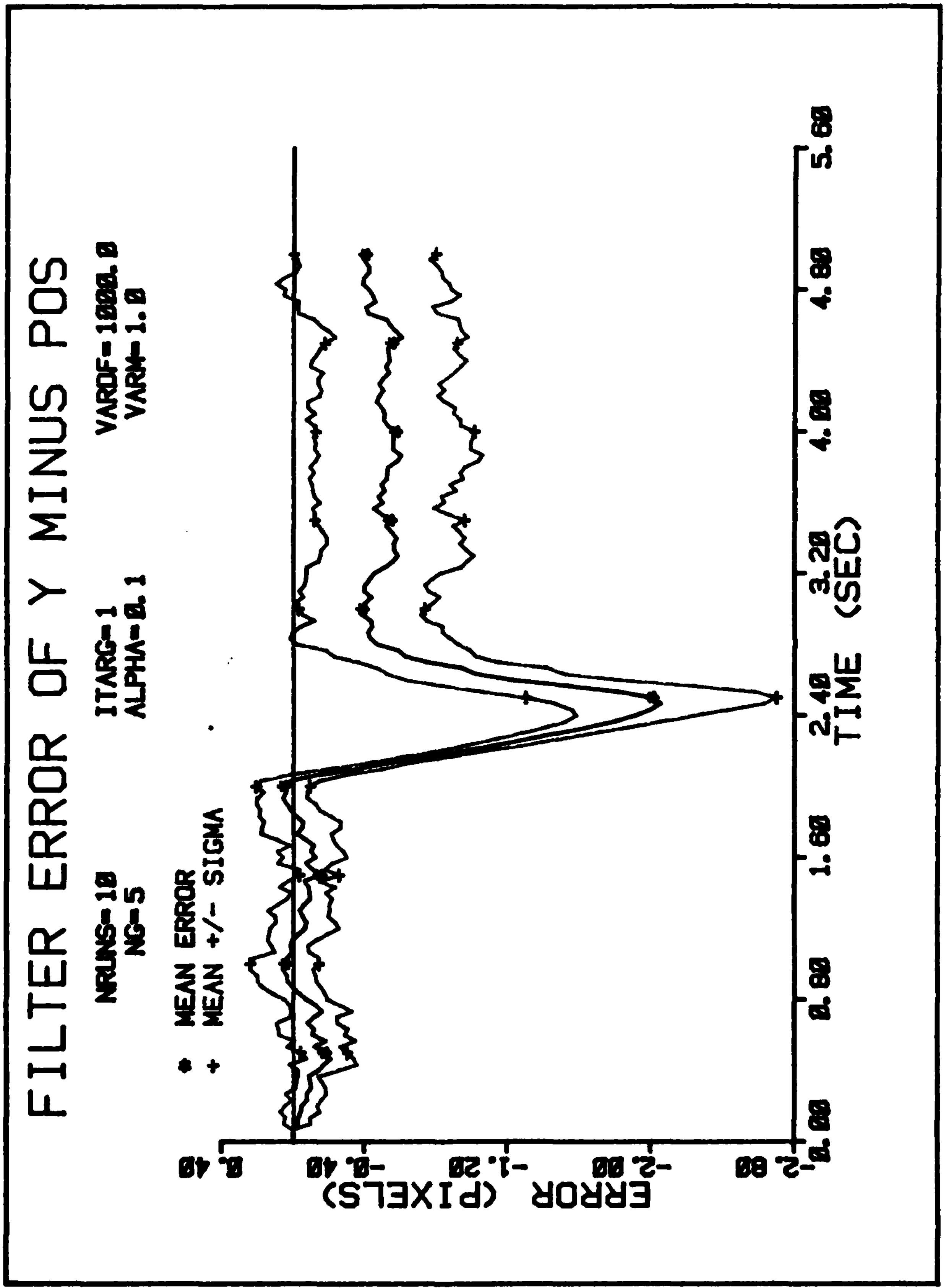


Figure E-61 Case 17 CTR Performance Plot

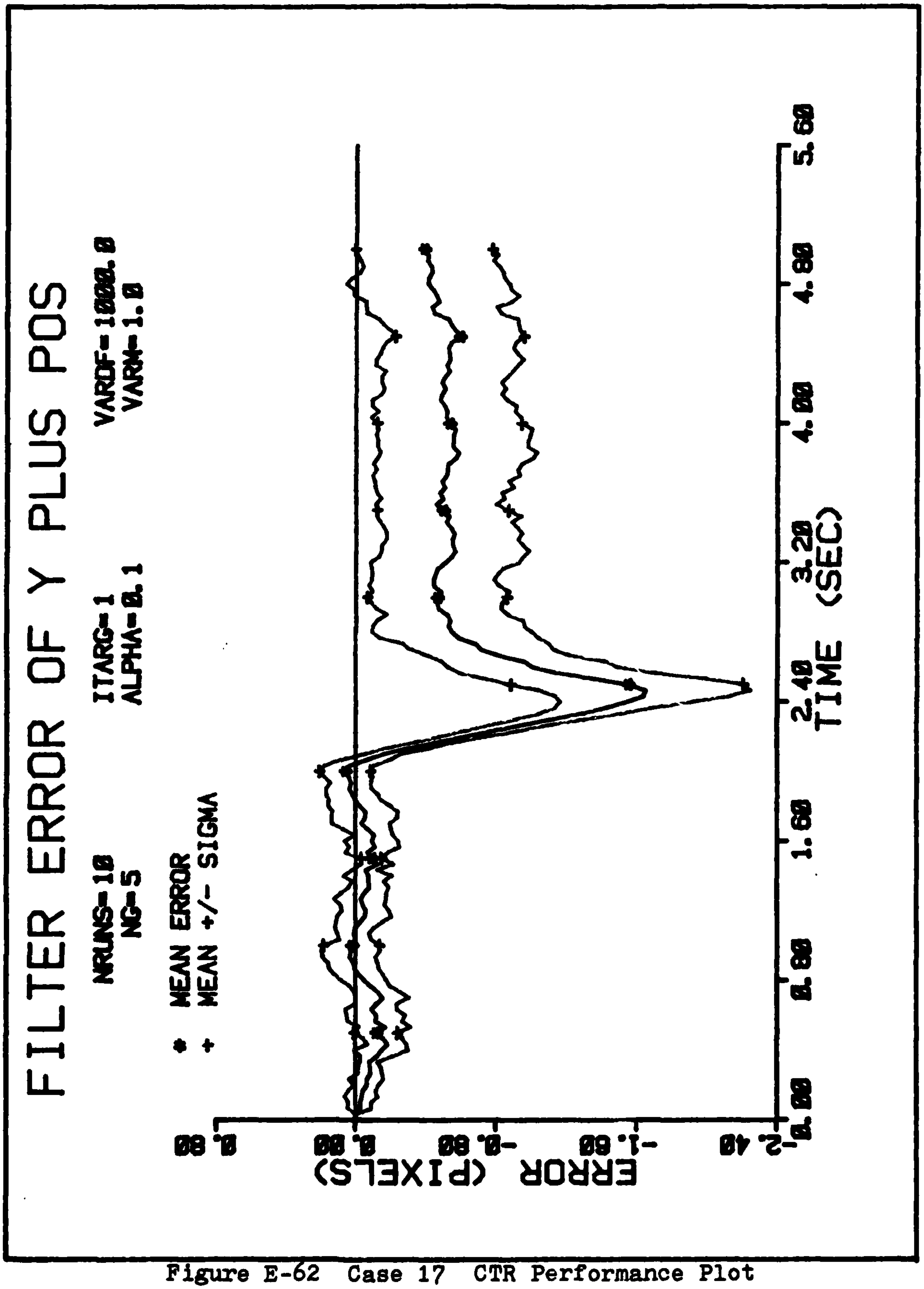


Figure E-62 Case 17 CTR Performance Plot

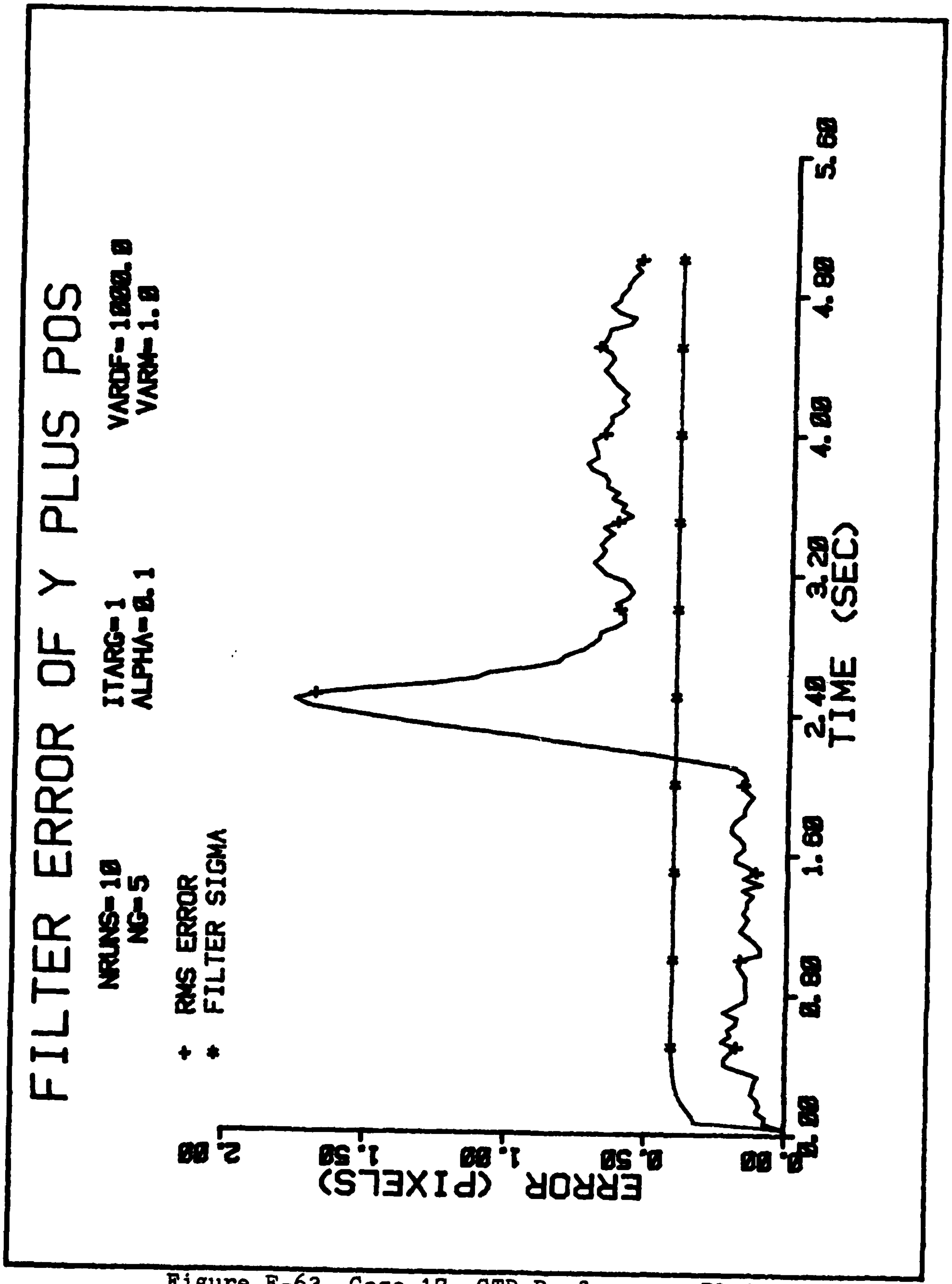


Figure E-63 Case 17 CTR Performance Plot

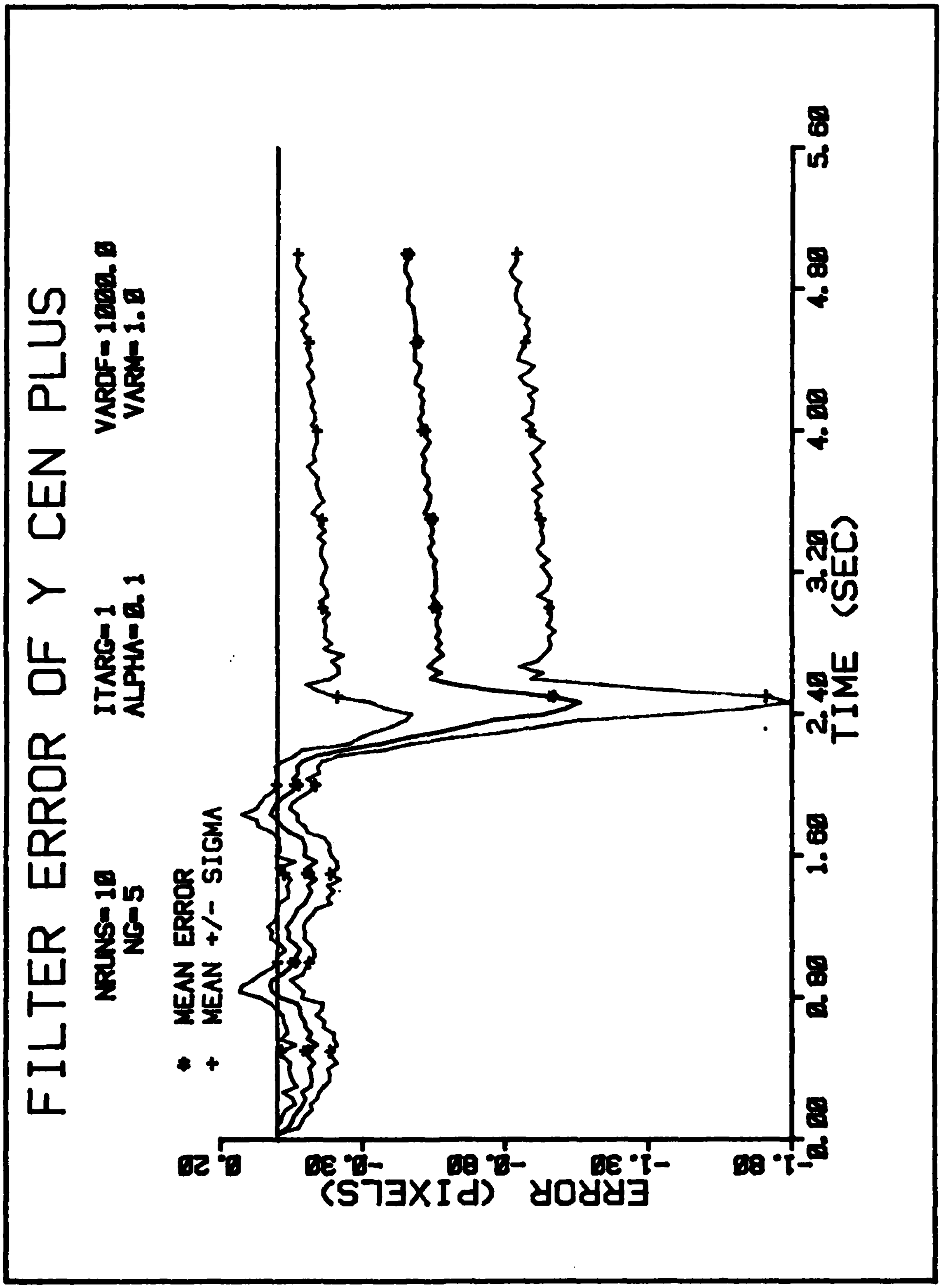


Figure E-64 Case 17 CTR Performance Plot

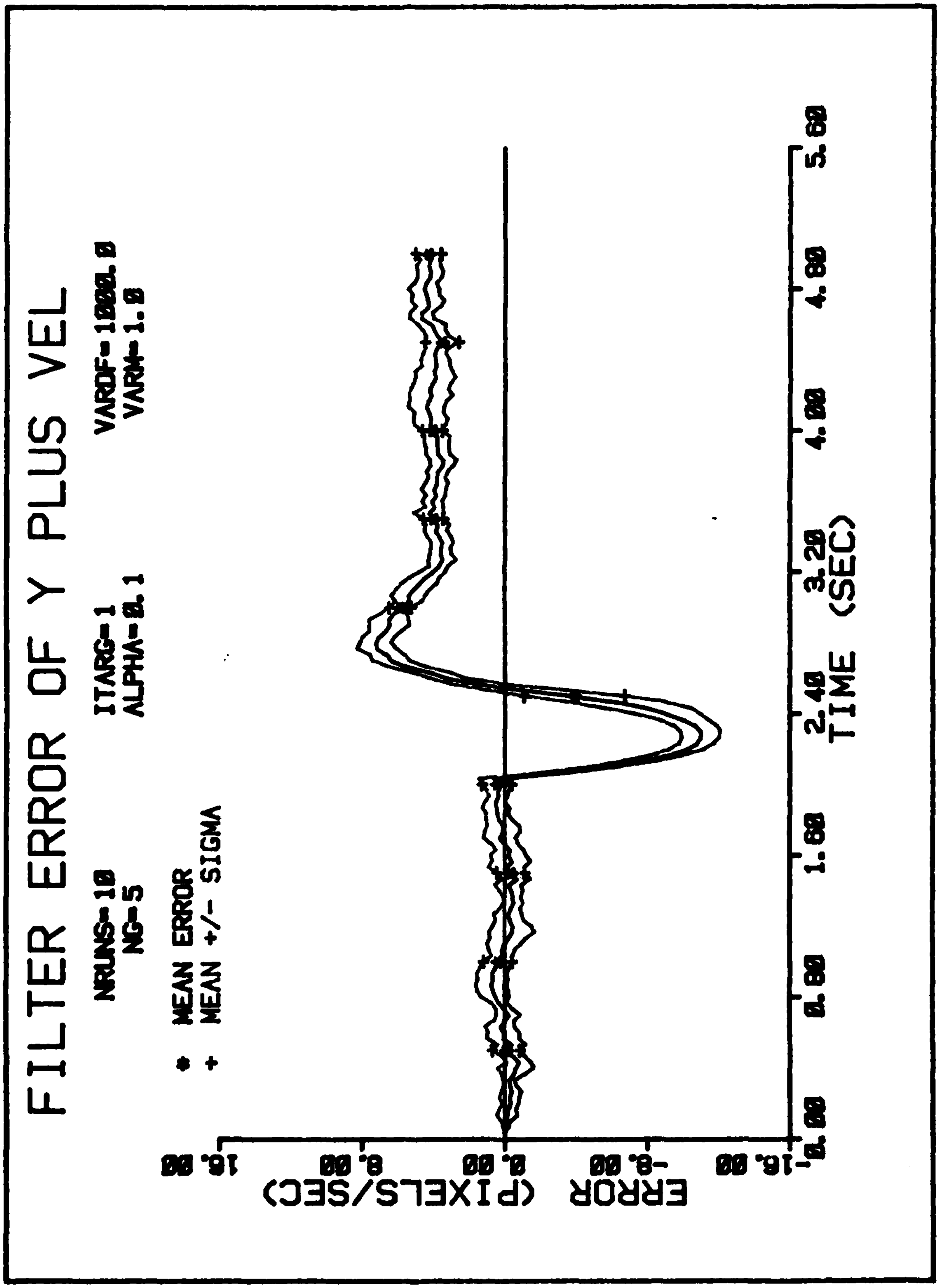


Figure E-65 Case 17 CTR Performance Plot

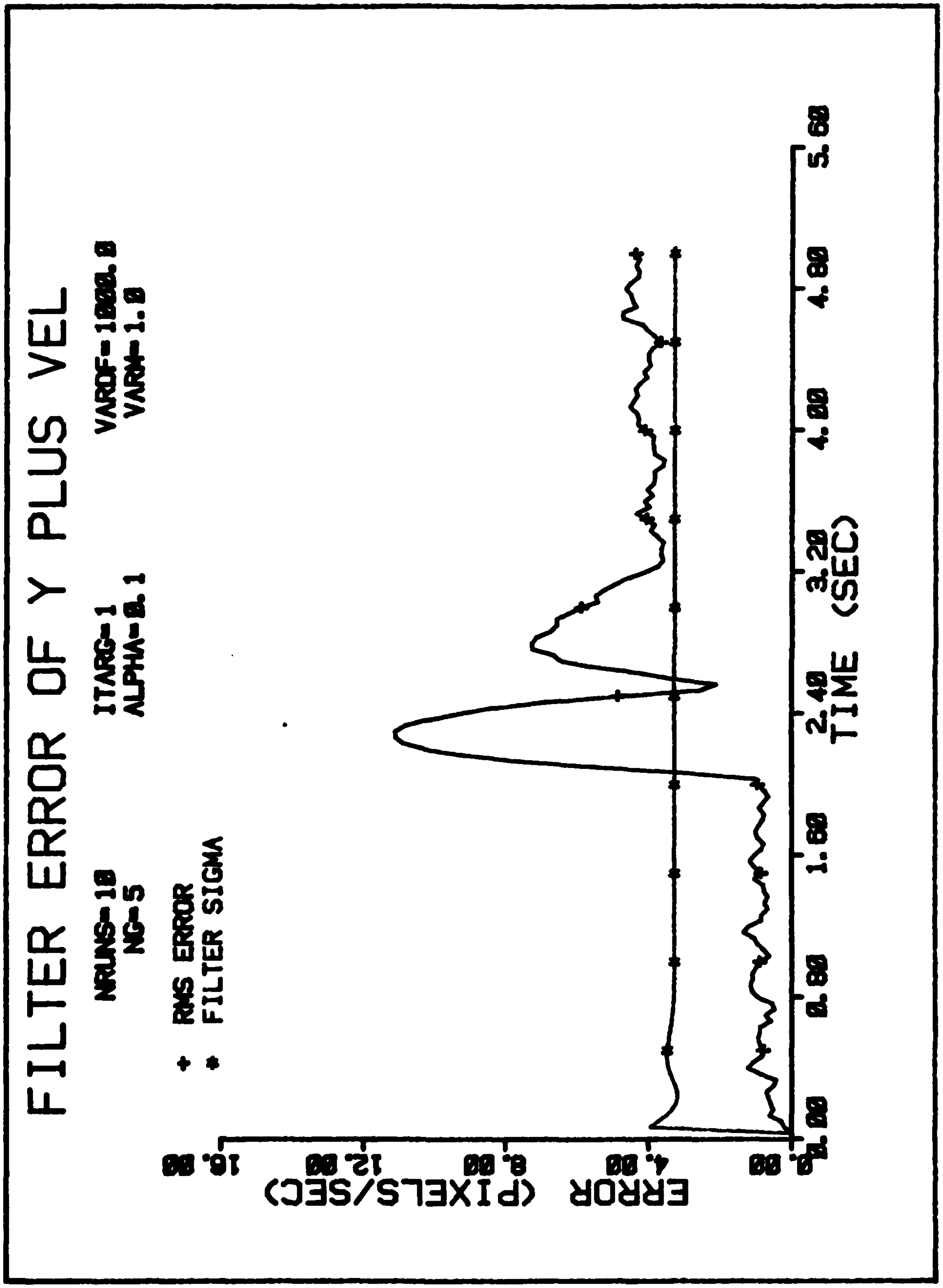


Figure E-66 Case 17 CTR Performance Plot

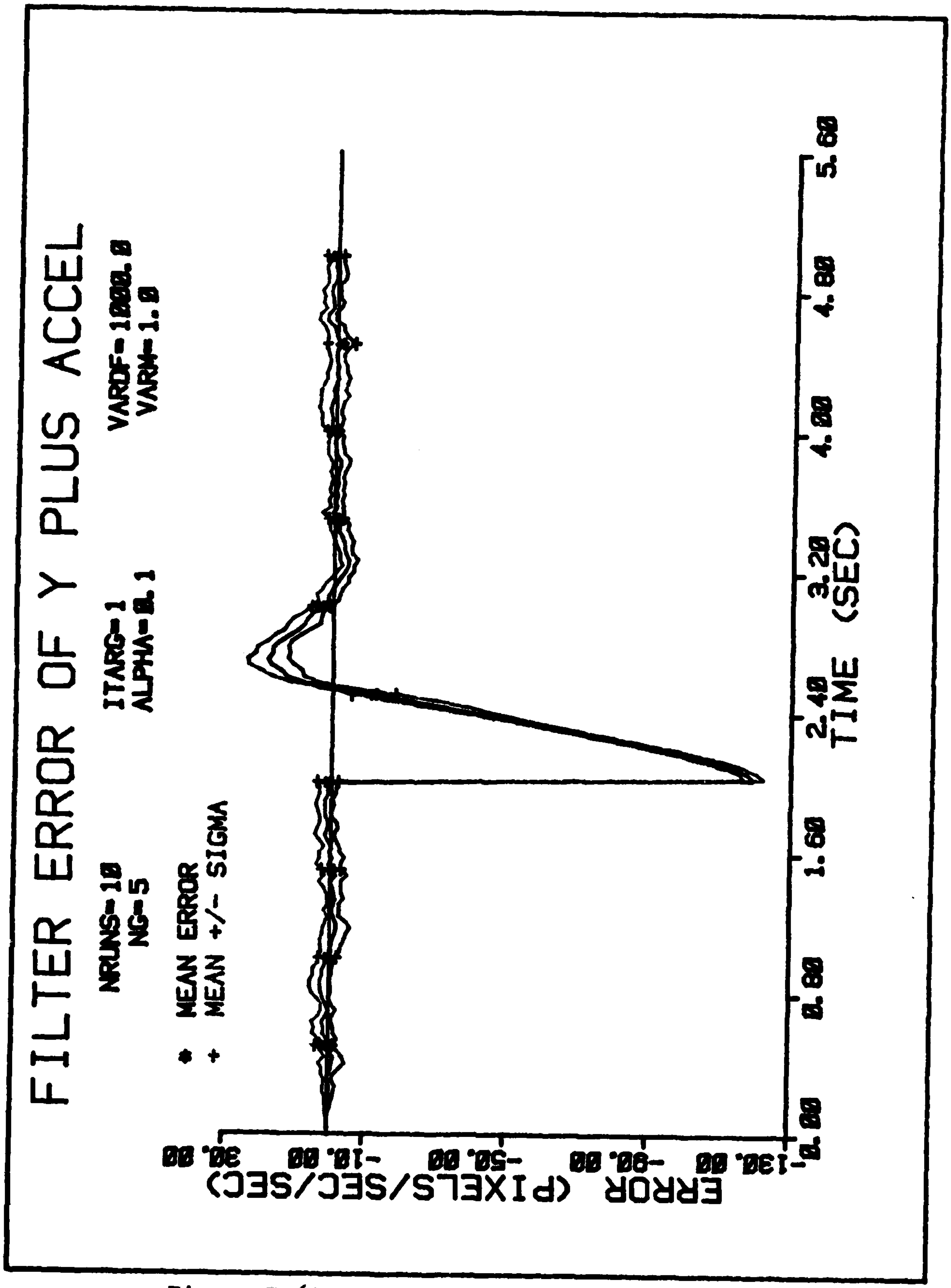


Figure E-67 Case 17 CTR Performance Plot

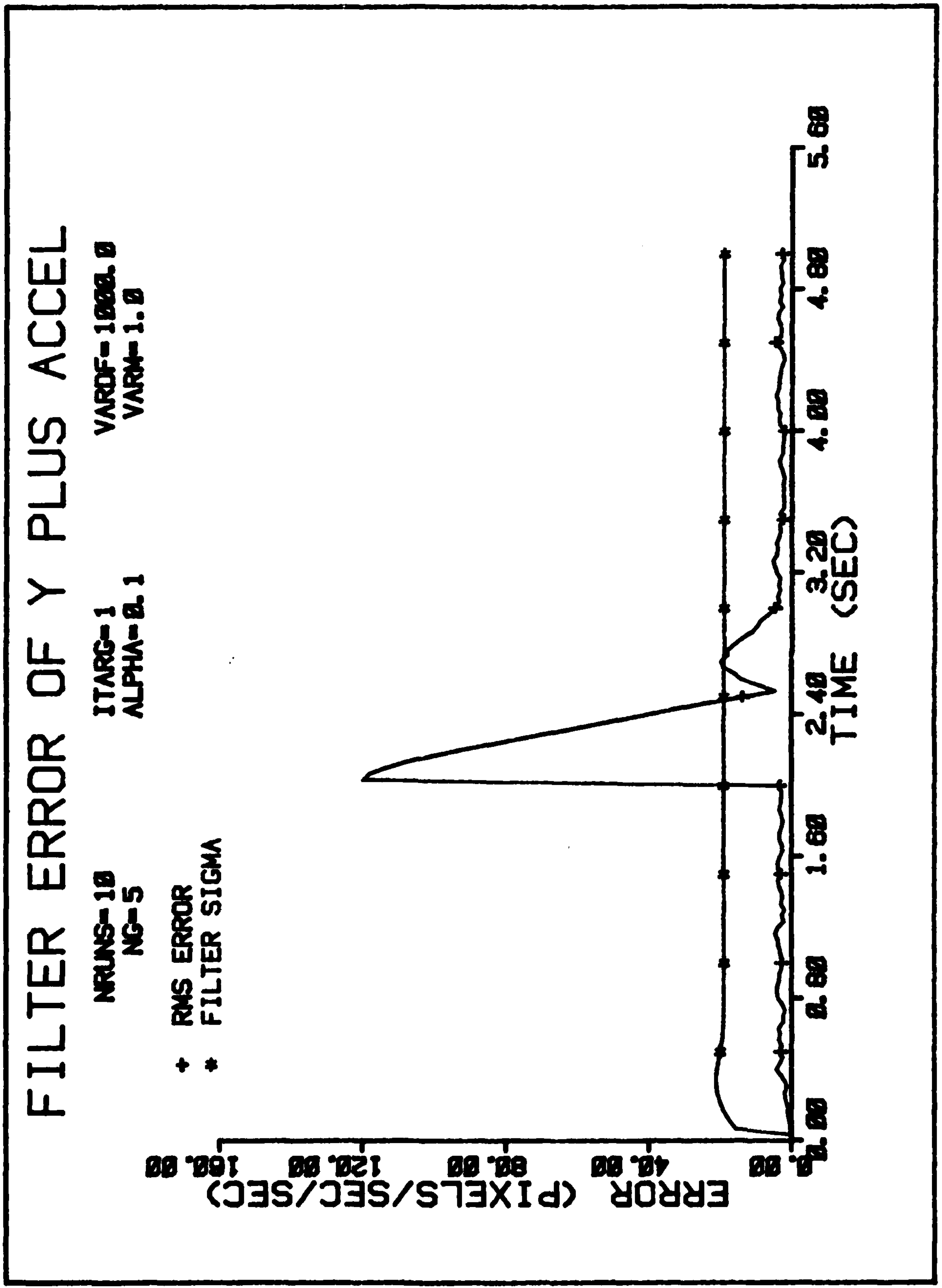


Figure E-68 Case 17 CTR Performance Plot

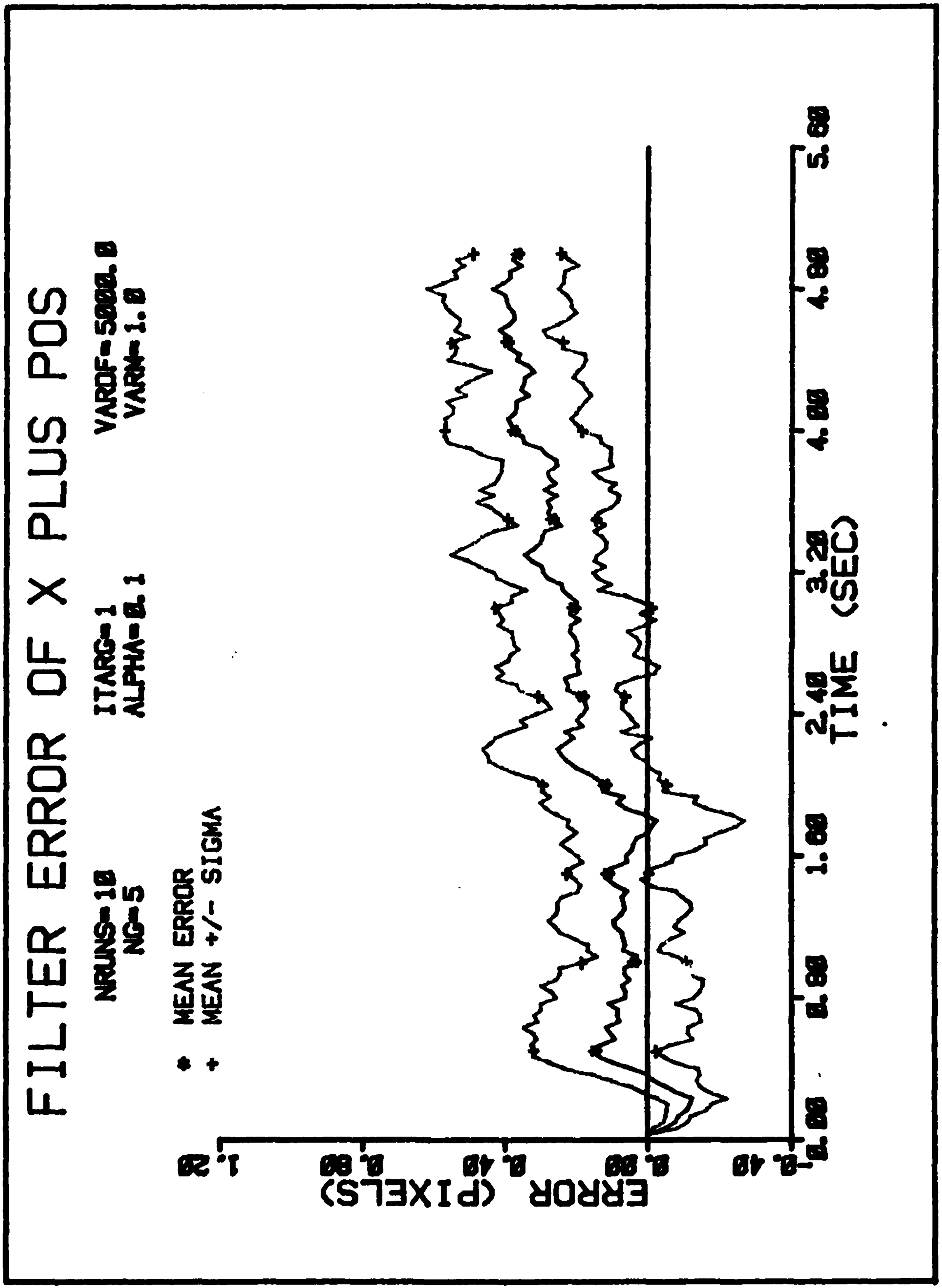


Figure E-69 Case 18 CTR Performance Plot

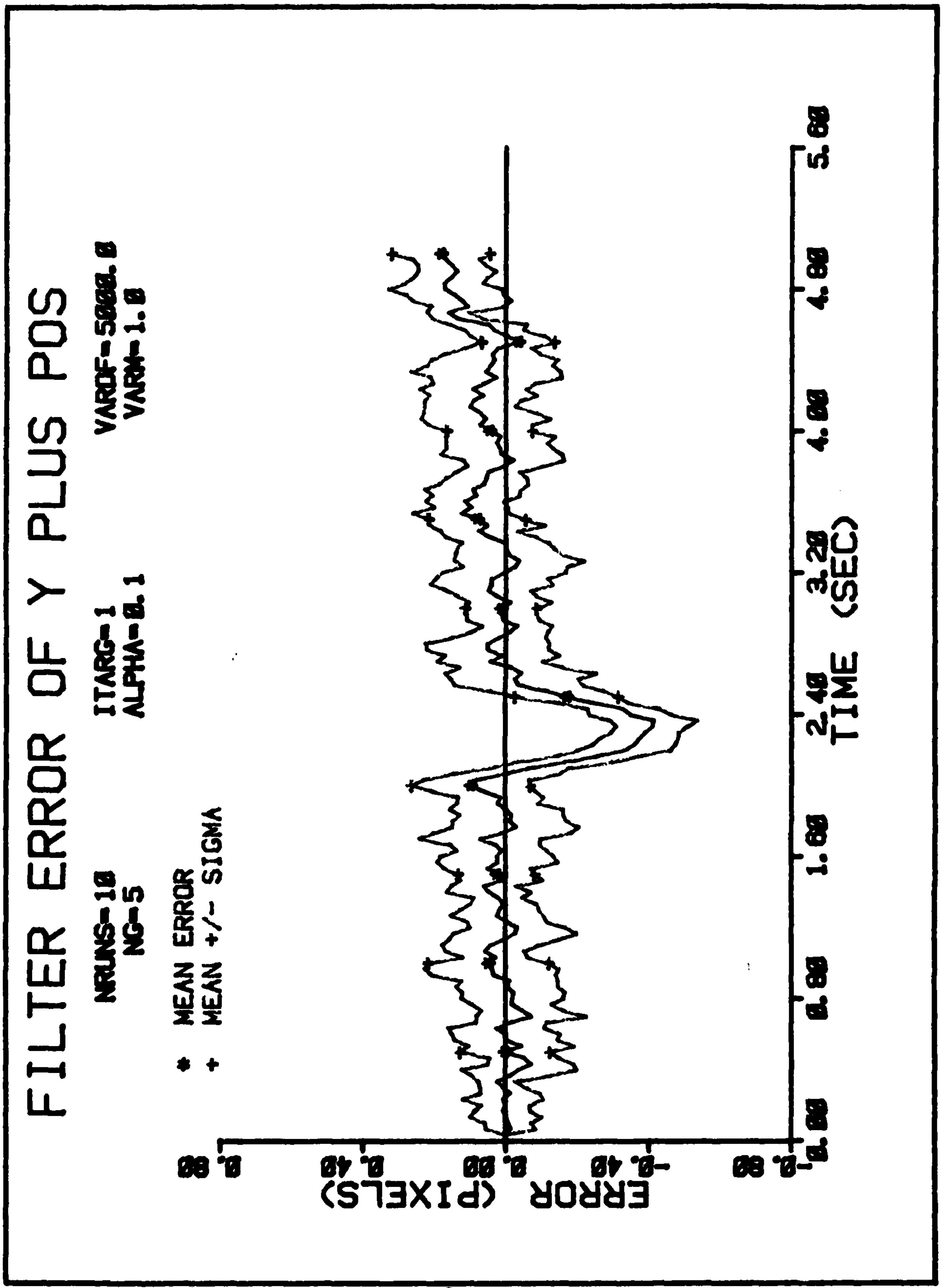


Figure E-70 Case 18 CTR Performance Plot

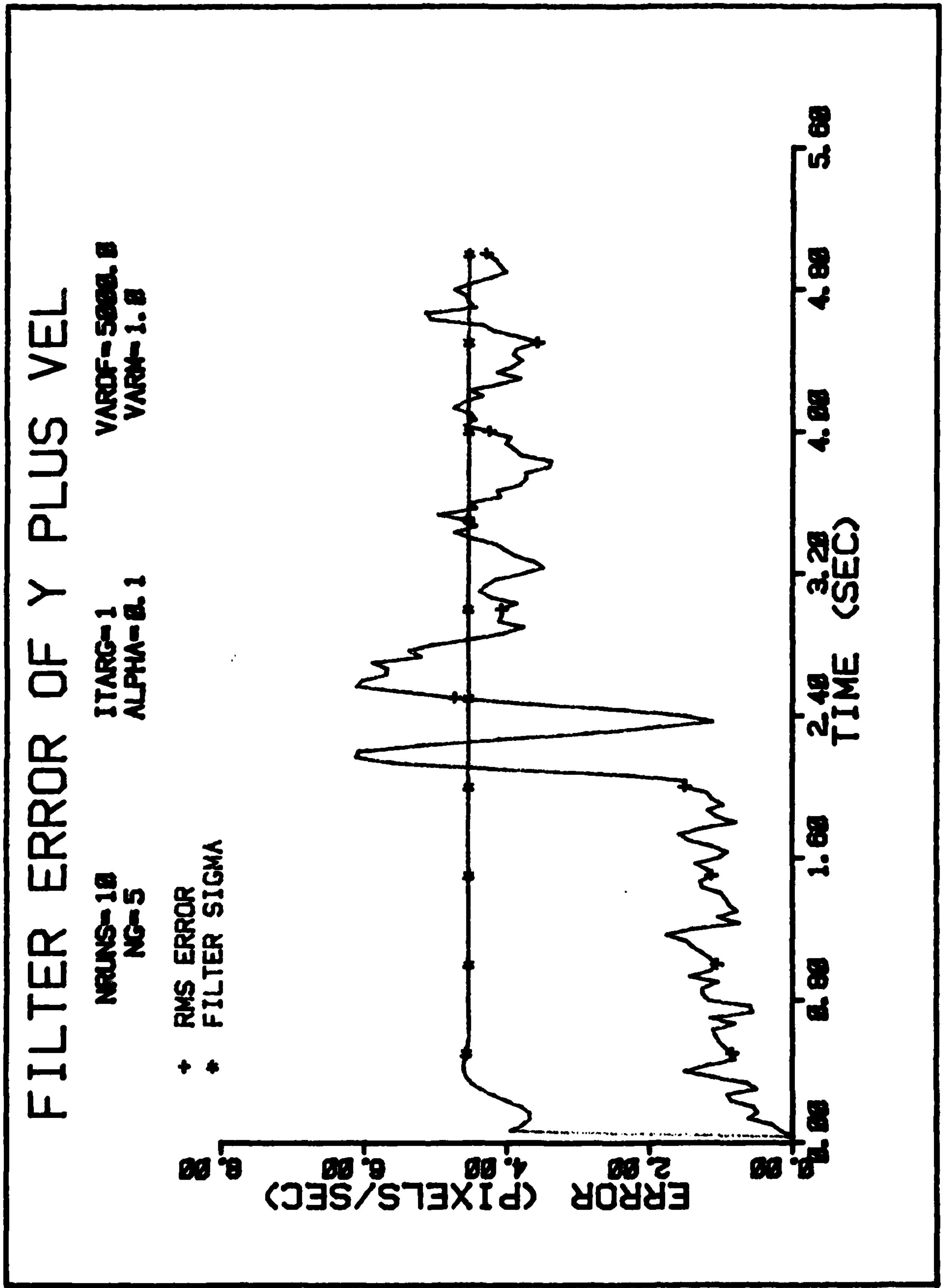


Figure E-71 Case 18 CTR Performance Plot

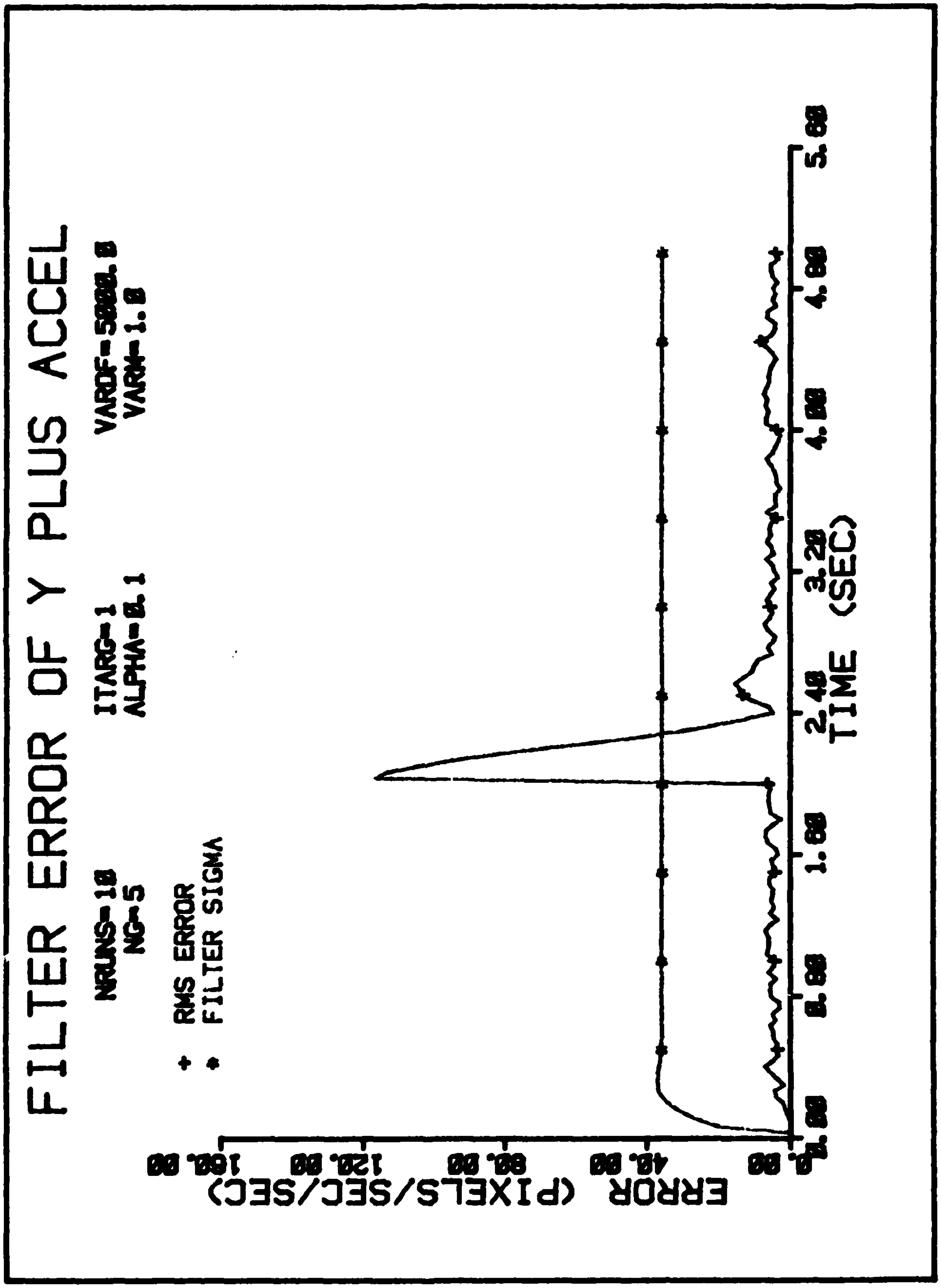


Figure E-72 Case 18 CTR Performance Plot

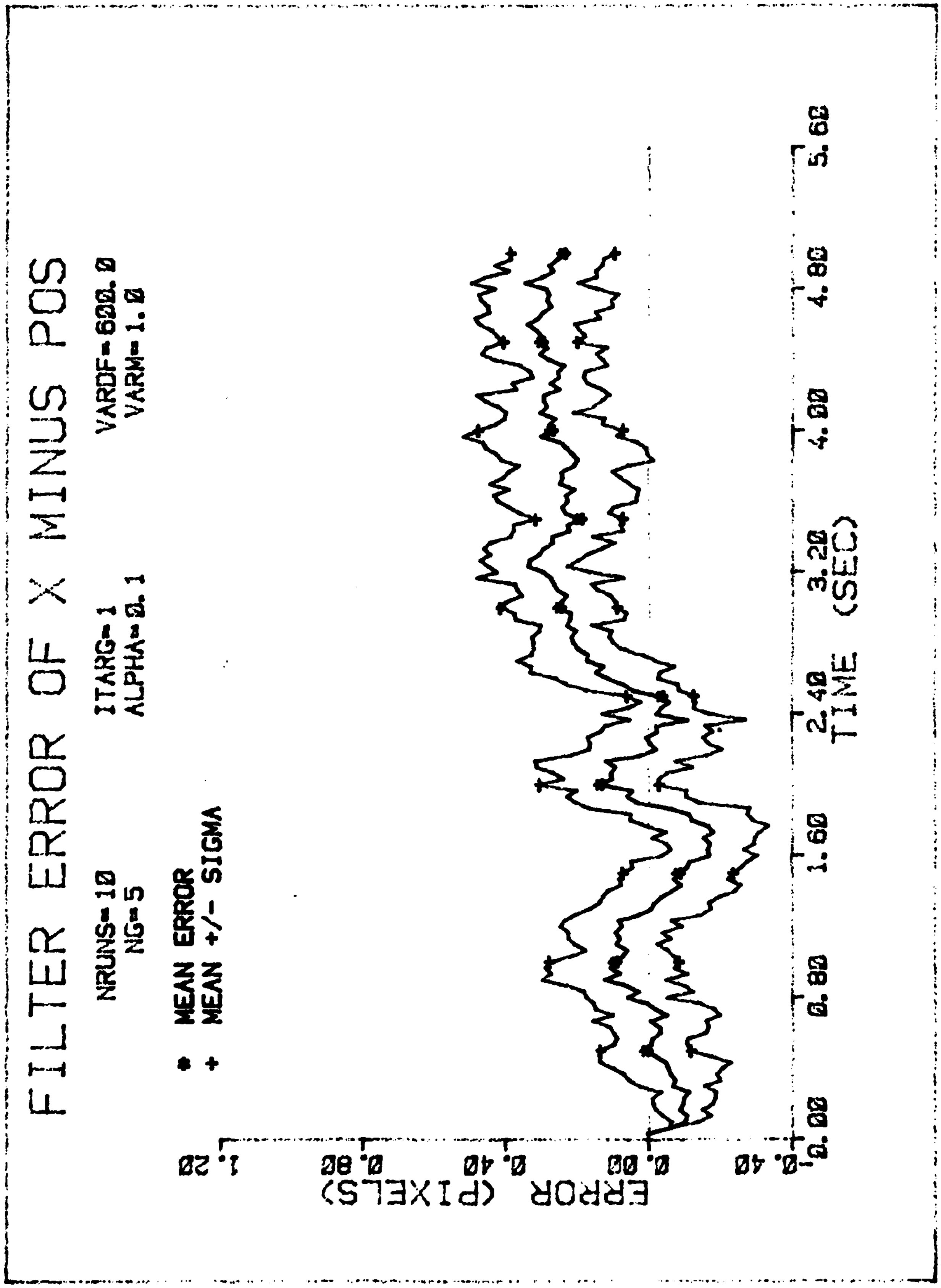


Figure E-73 Case 19 CTR Performance Plot

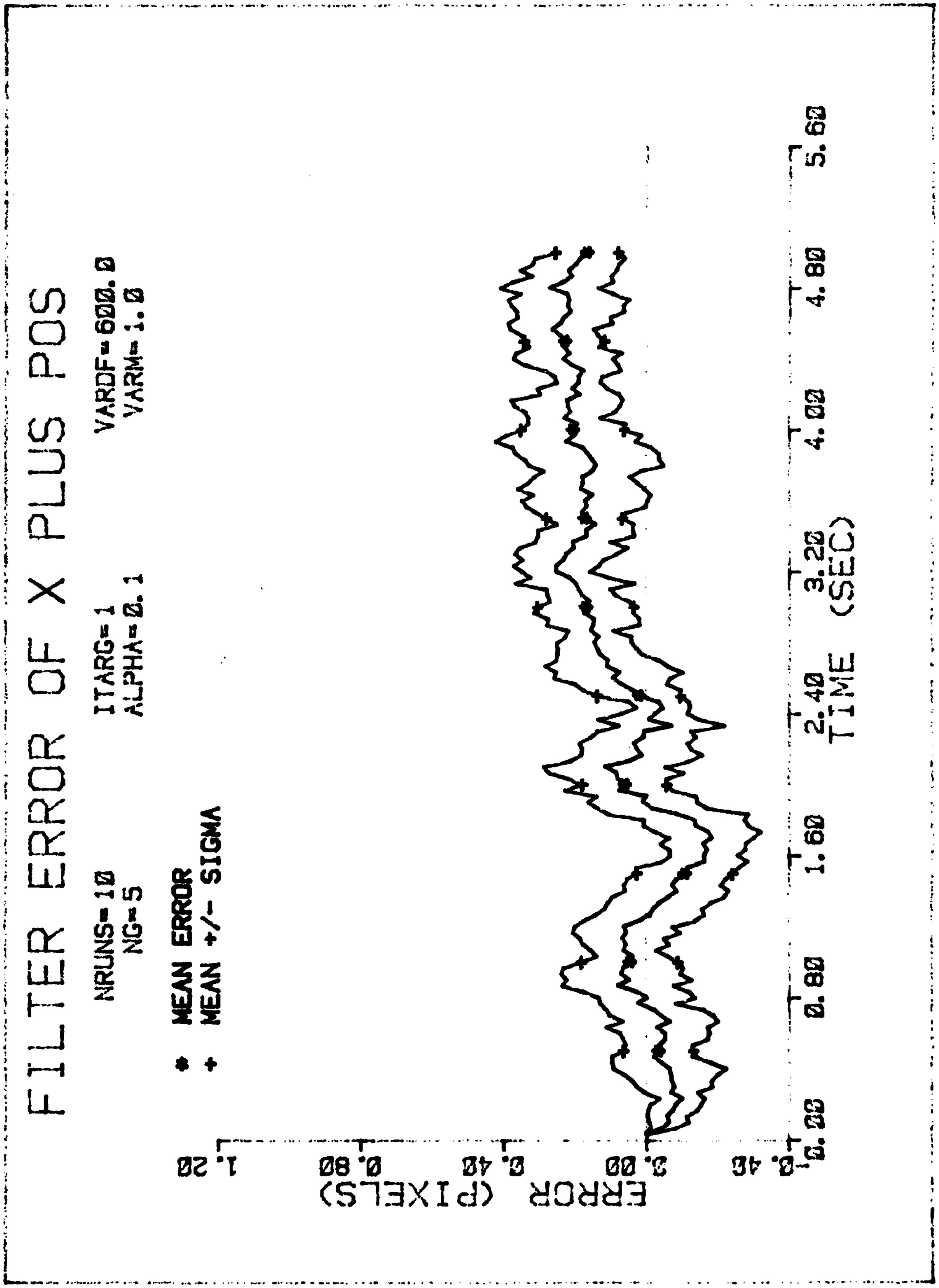


Figure E-74 Case 19 CTR Performance Plot

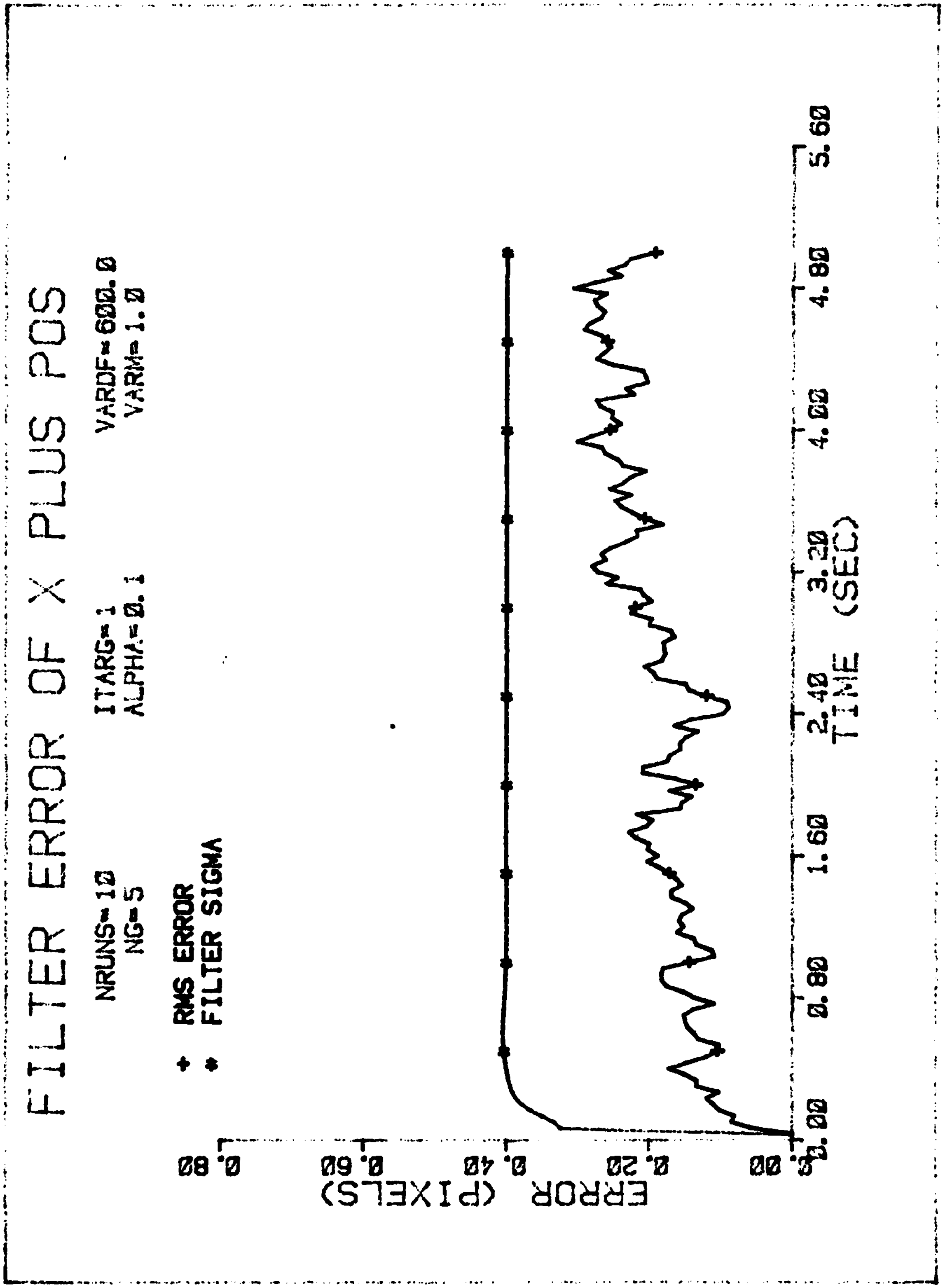


Figure E-75 Case 19 CTR Performance Plot

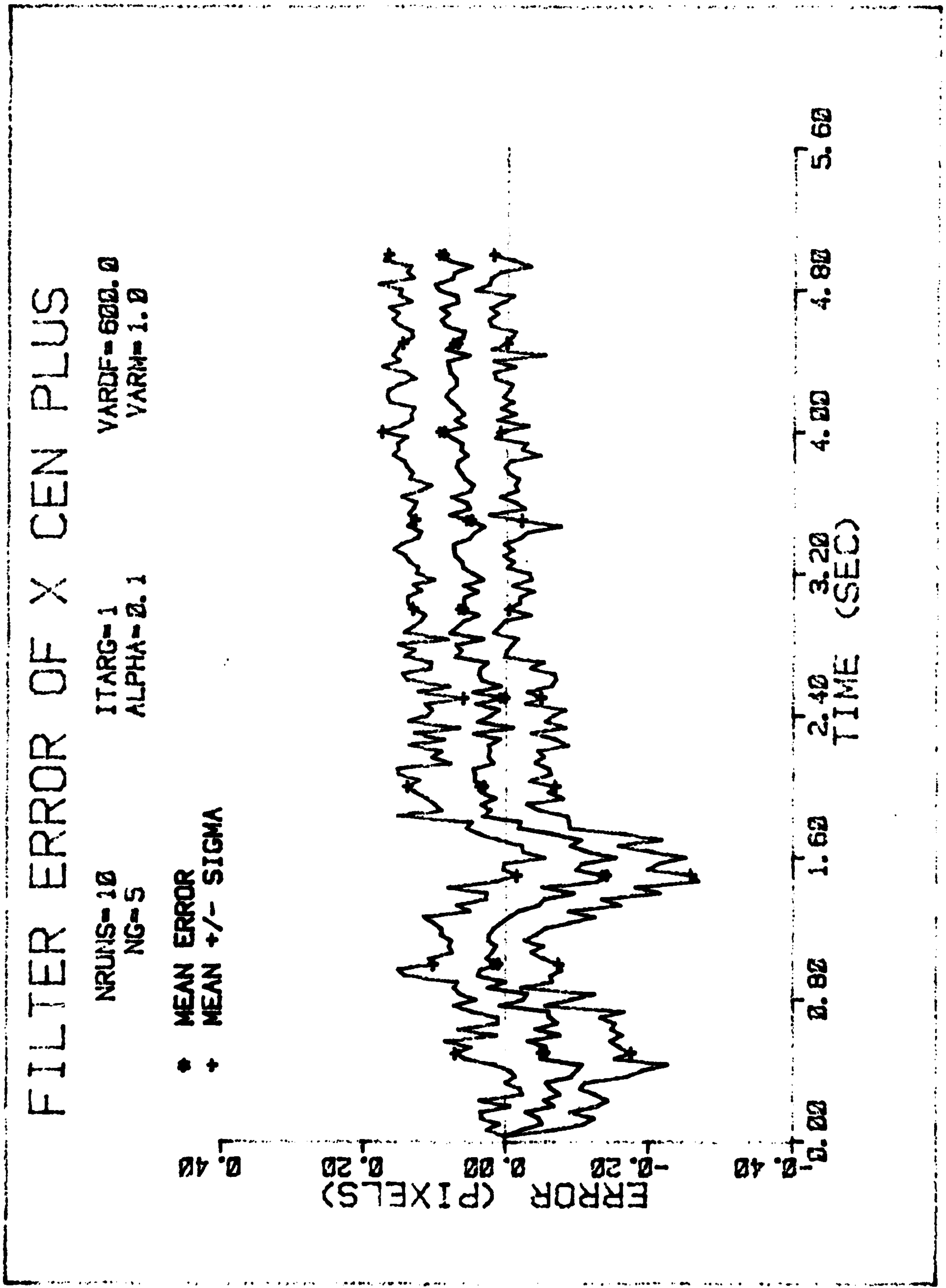


Figure E-76 Case 19 CTR Performance Plot

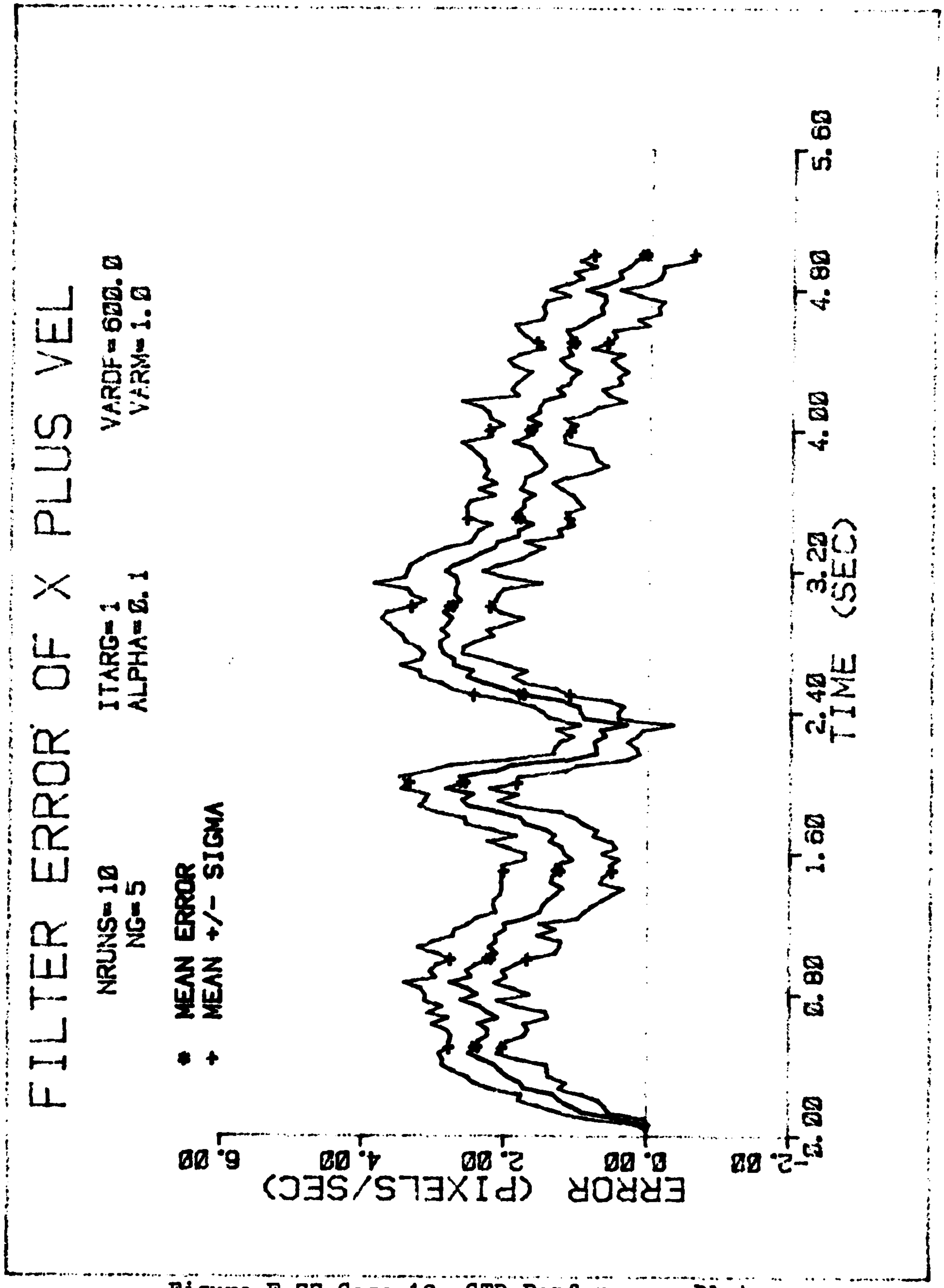


Figure E-77 Case 19 CTR Performance Plot

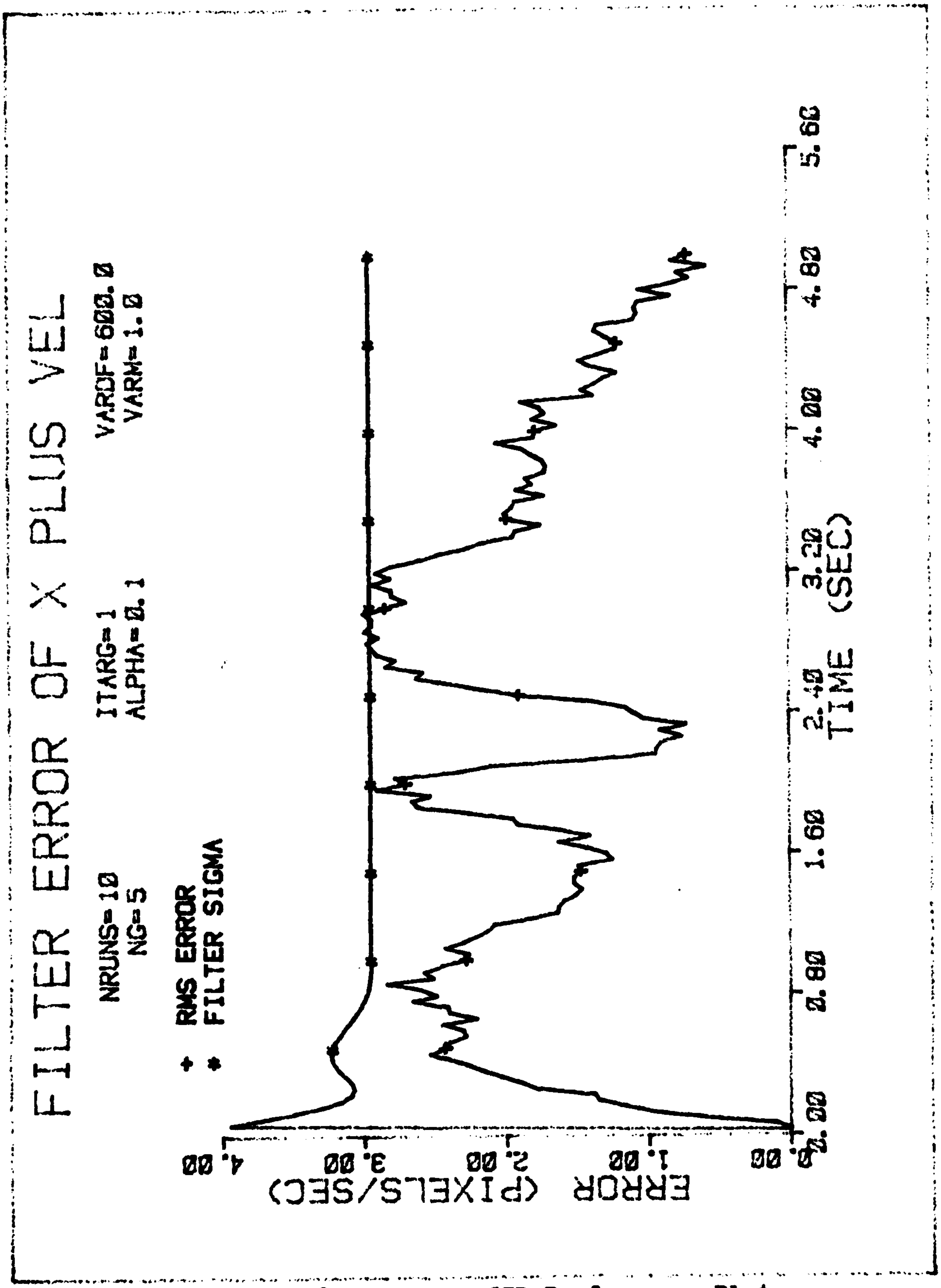


Figure E-78 Case 19 CTR Performance Plot

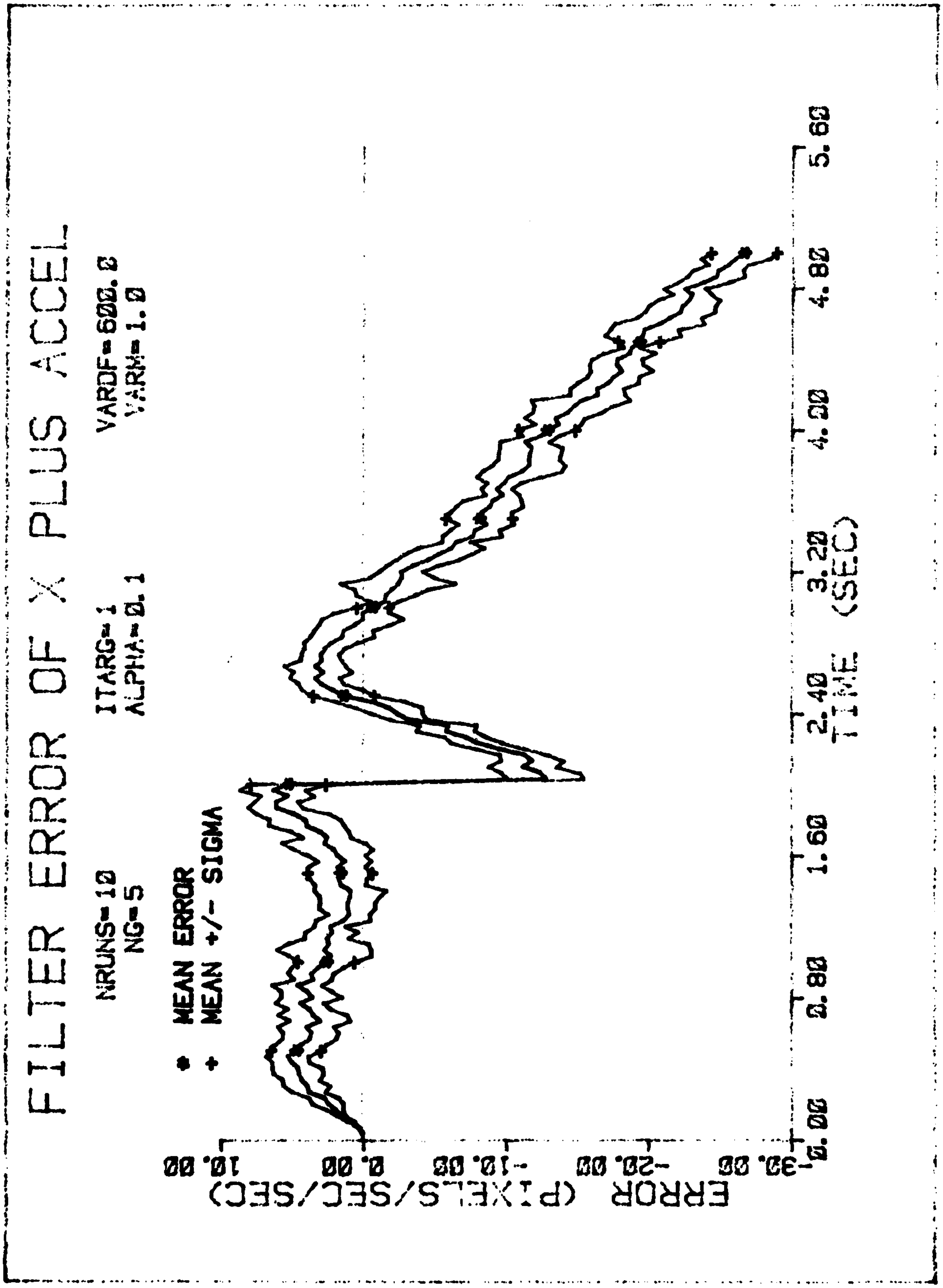


Figure E-79 Case 19 CTR Performance Plot

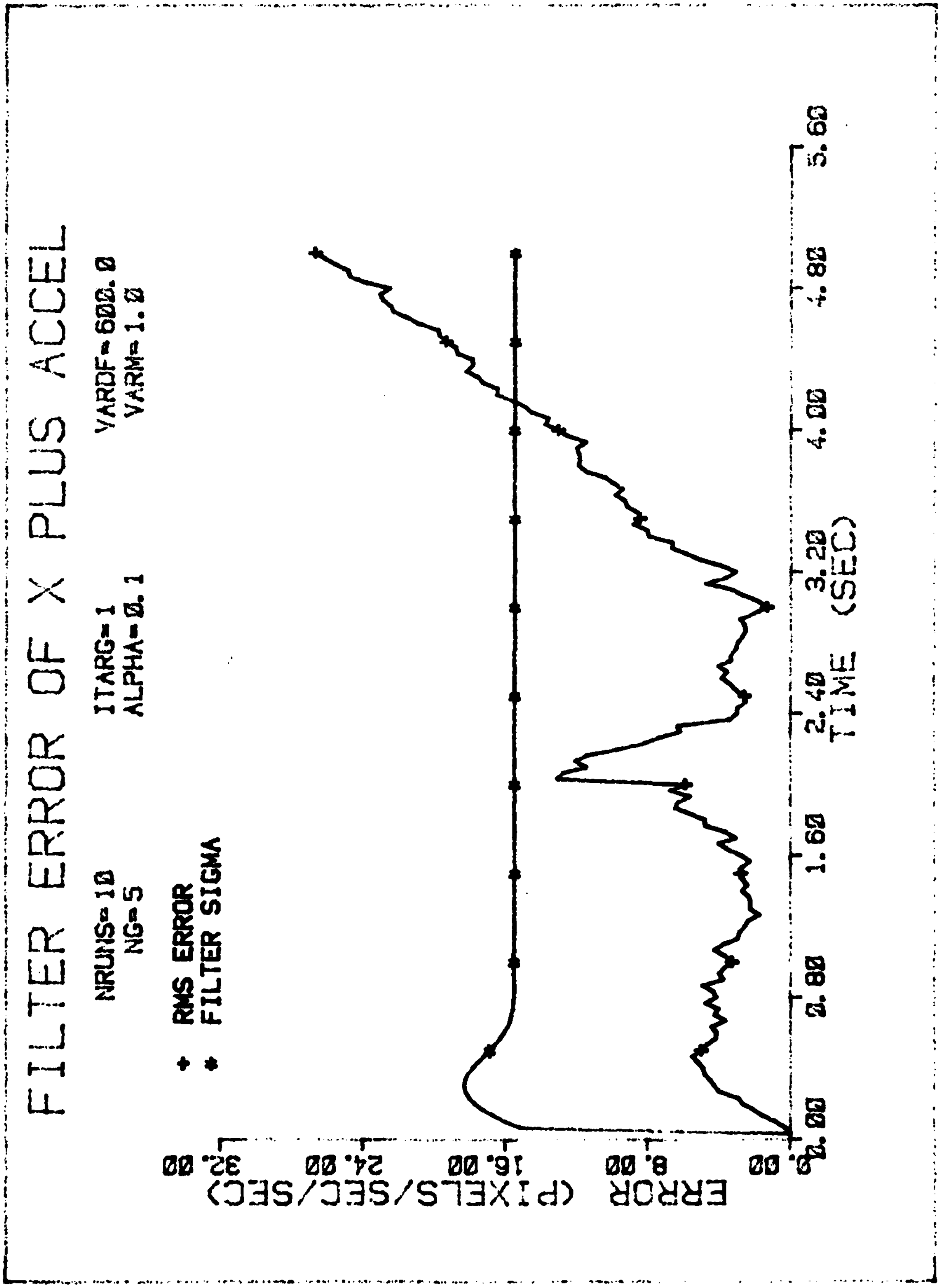


Figure E-80 Case 19 CTR Performance Plot

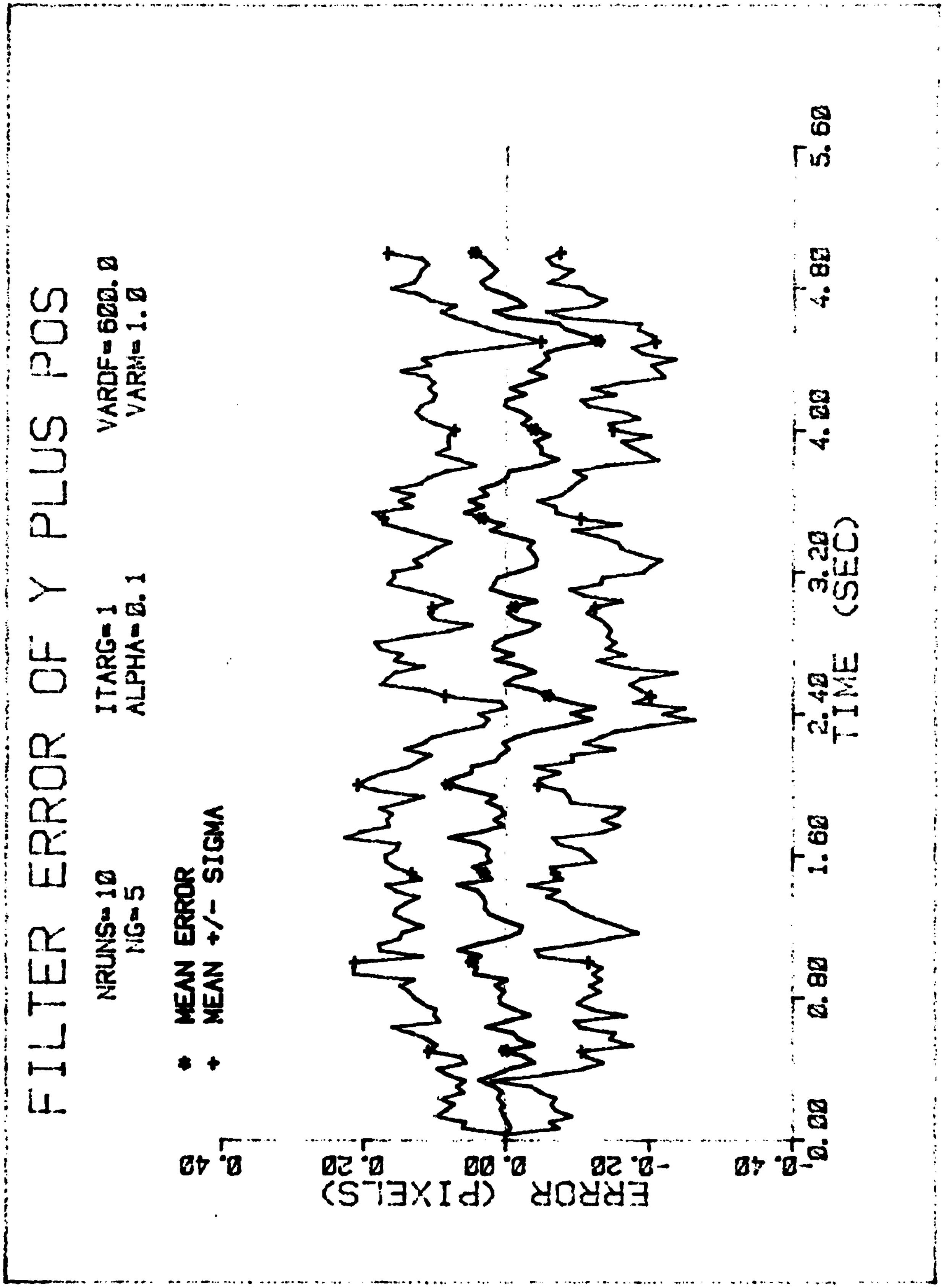


Figure E-81 Case 19 CTR Performance Plot

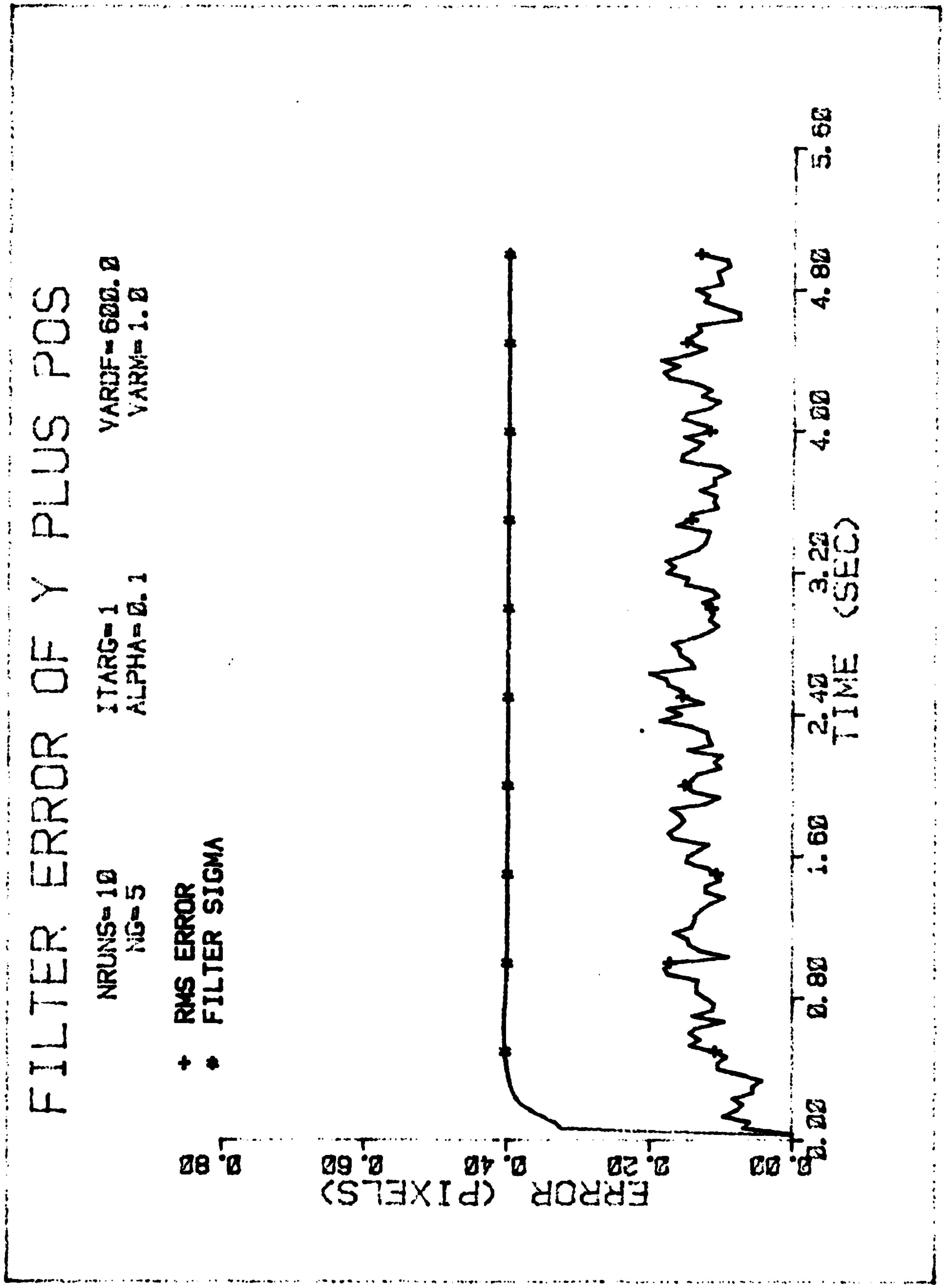


Figure E-82 Case 19 CTR Performance Plot

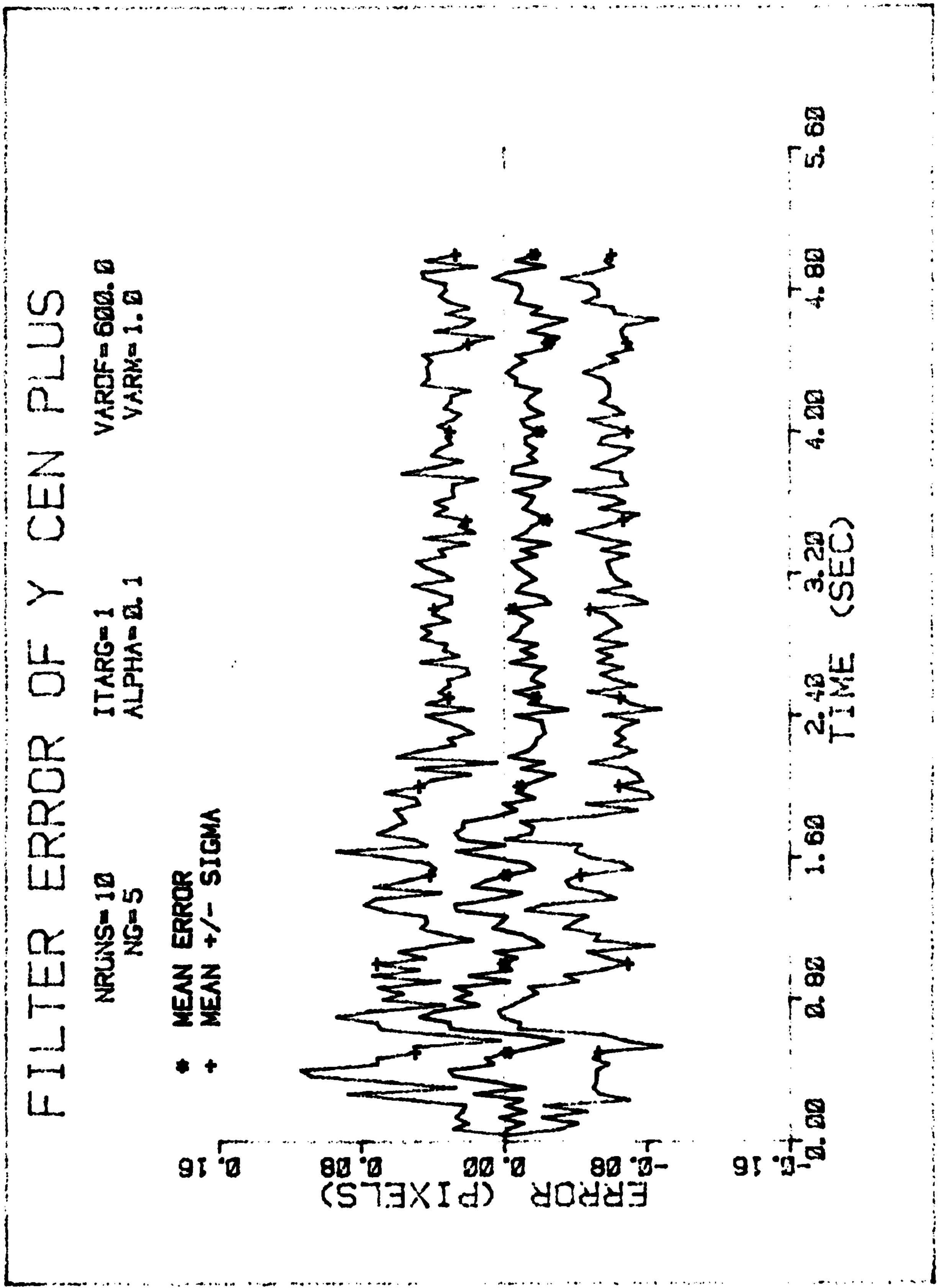


Figure E-83 Case 19 CTR Performance Plot

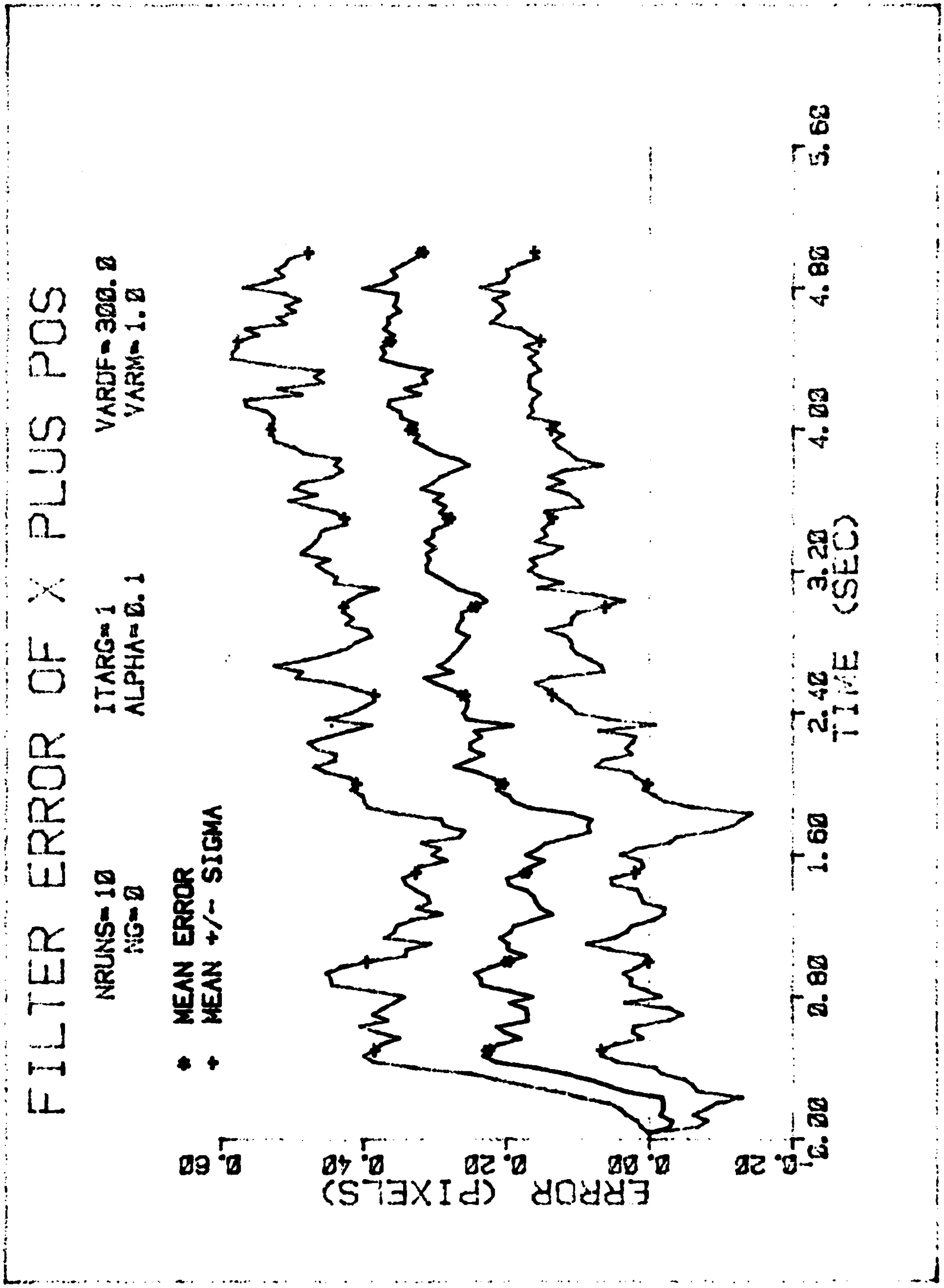


Figure E-84 Case 20 CTR Performance Plot

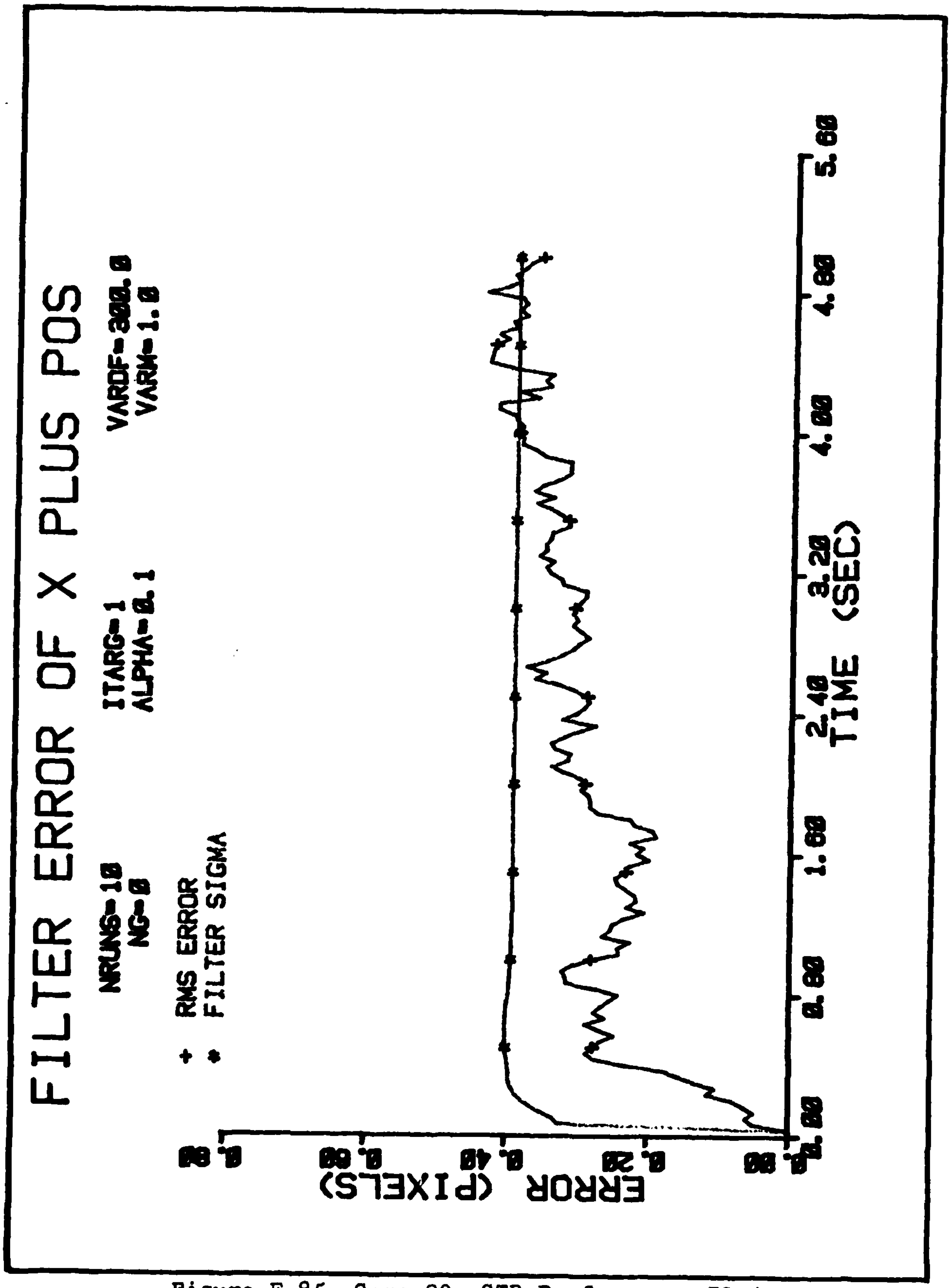


Figure E-85 Case 20 CTR Performance Plot

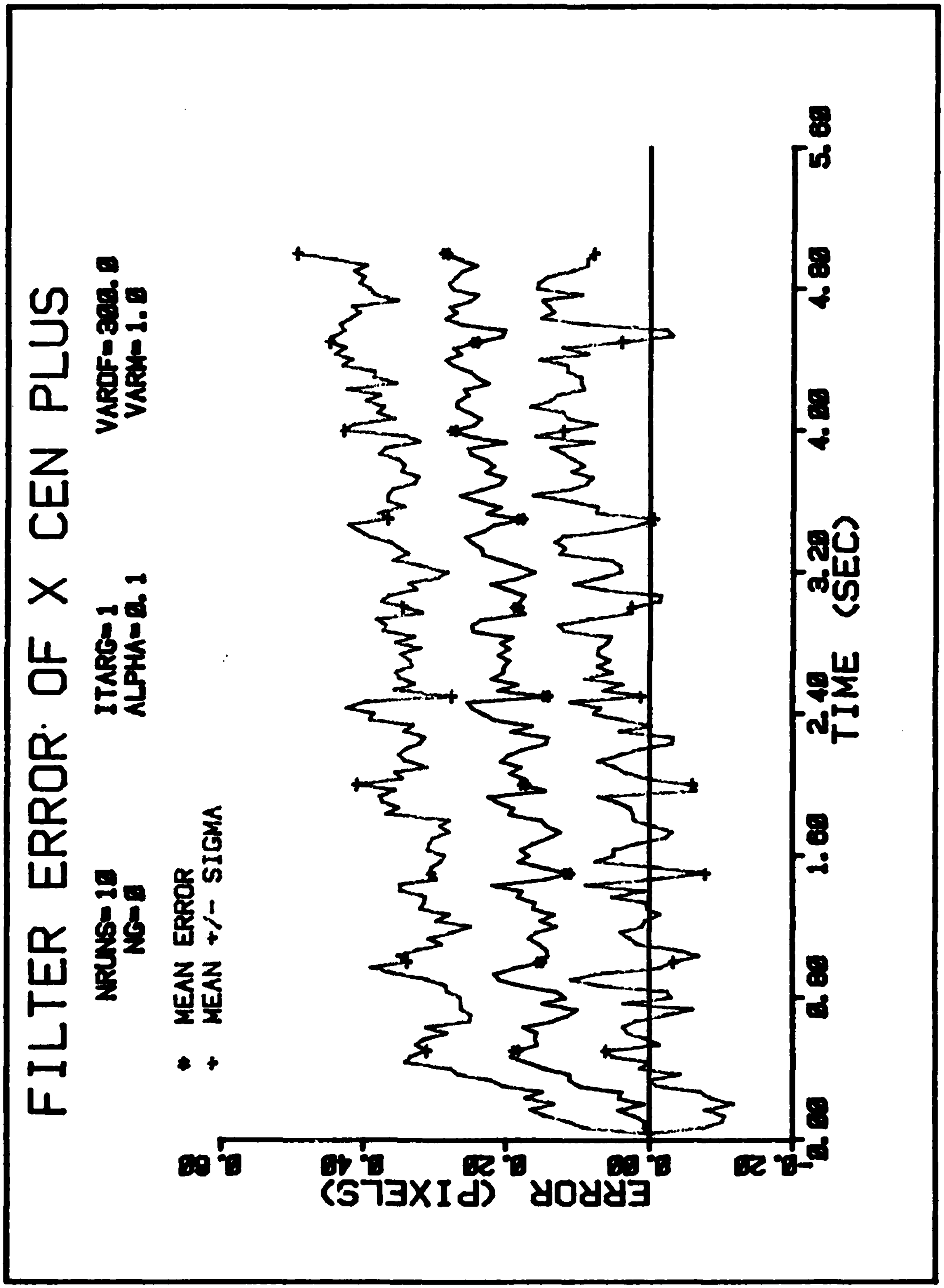


Figure E-86 Case 20 CTR Performance Plot

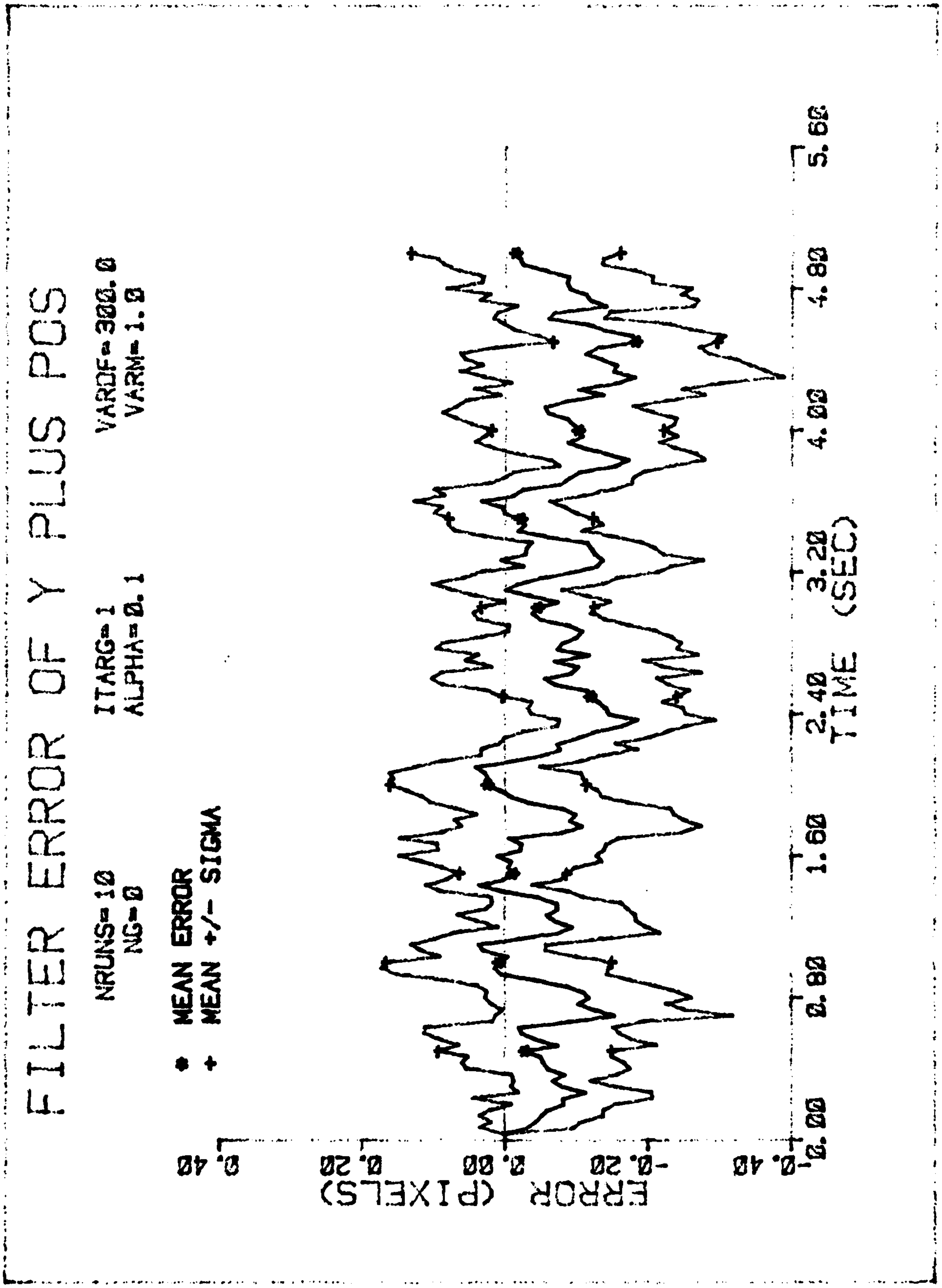


Figure E-87 Case 20 CTR Performance Plot

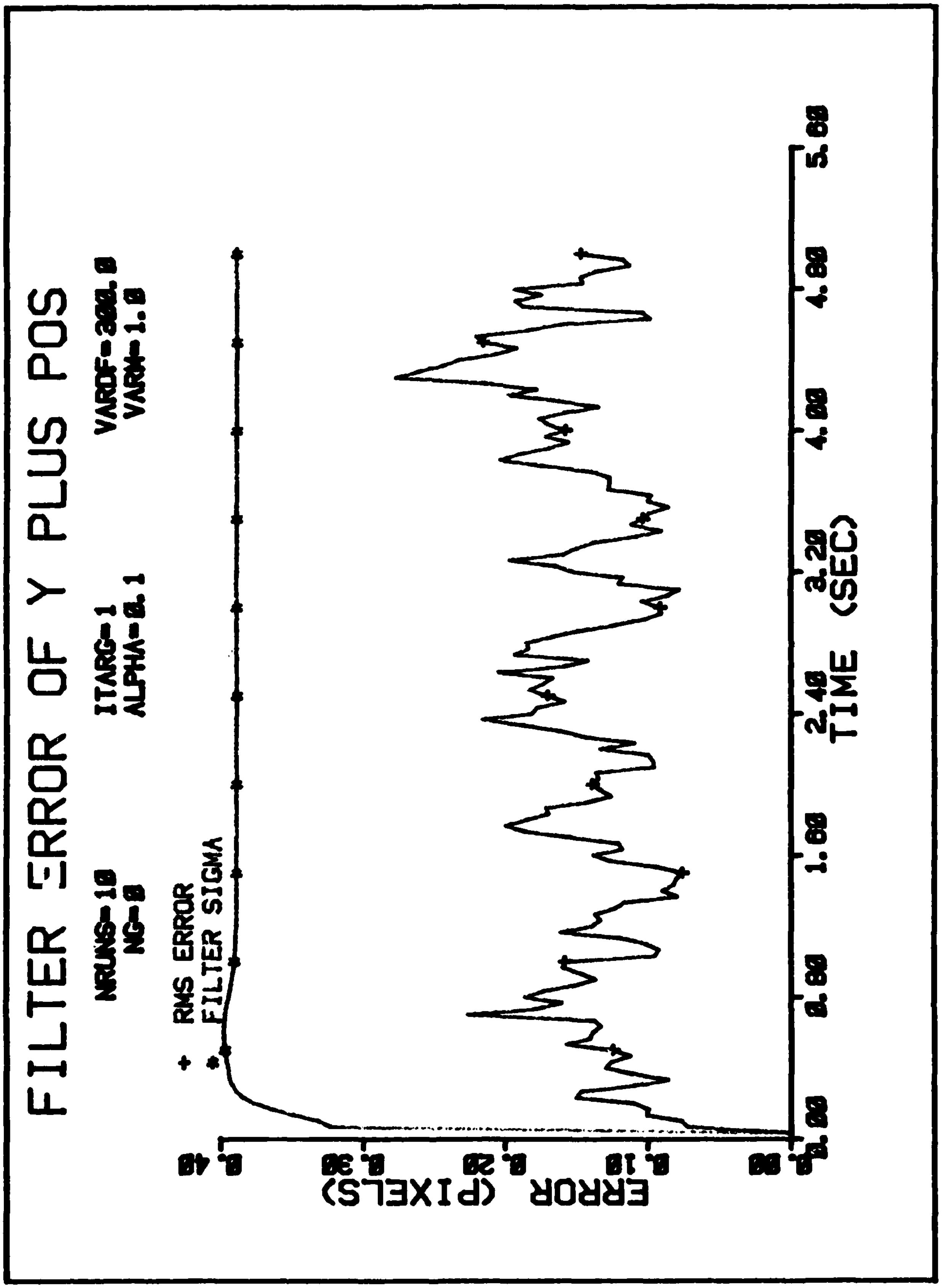


Figure E-88 Case 20 CTR Performance Plot

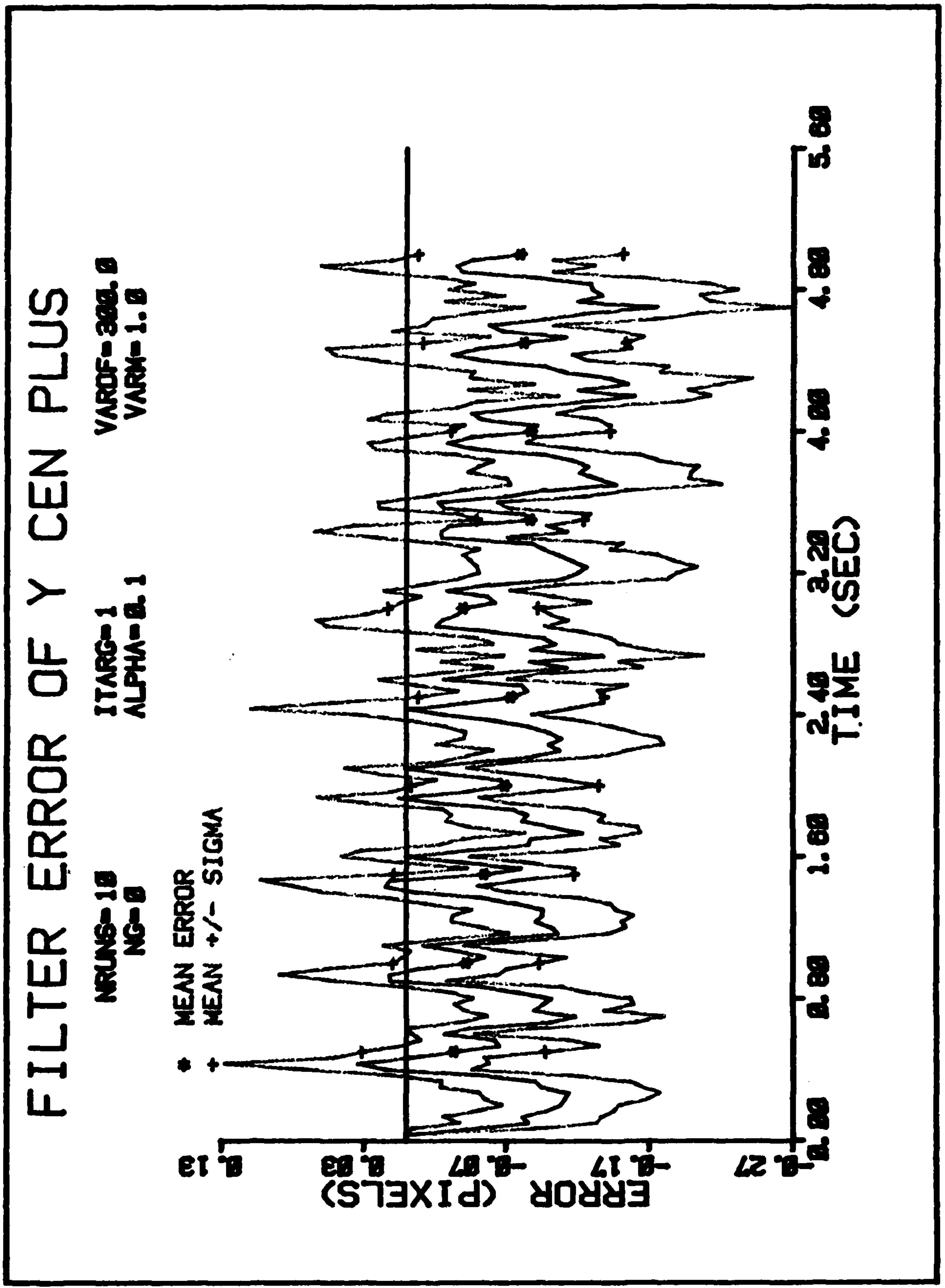


Figure E-89 Case 20 CTR Performance Plot

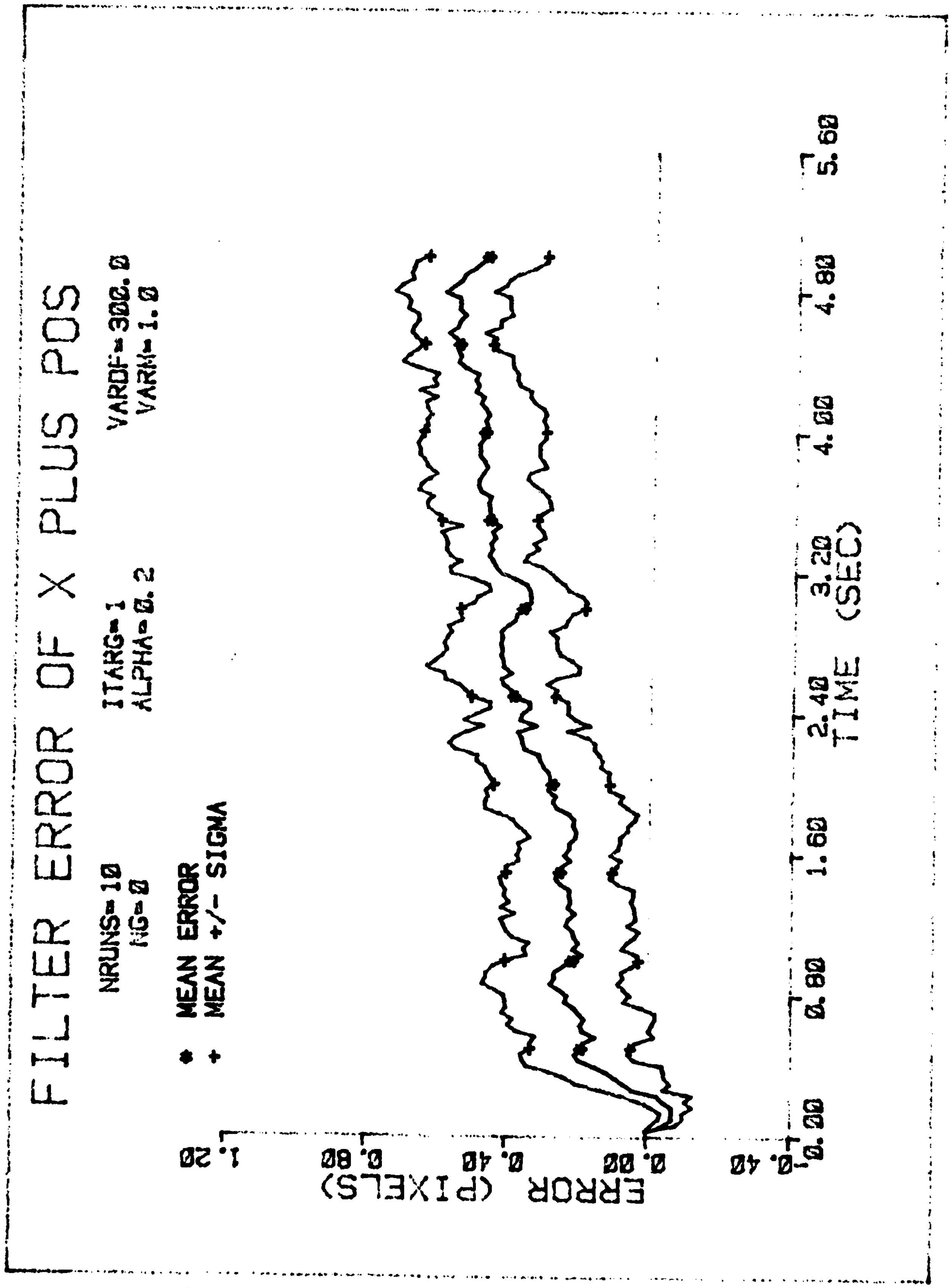


Figure E-90 Case 21 CTR Performance Plot

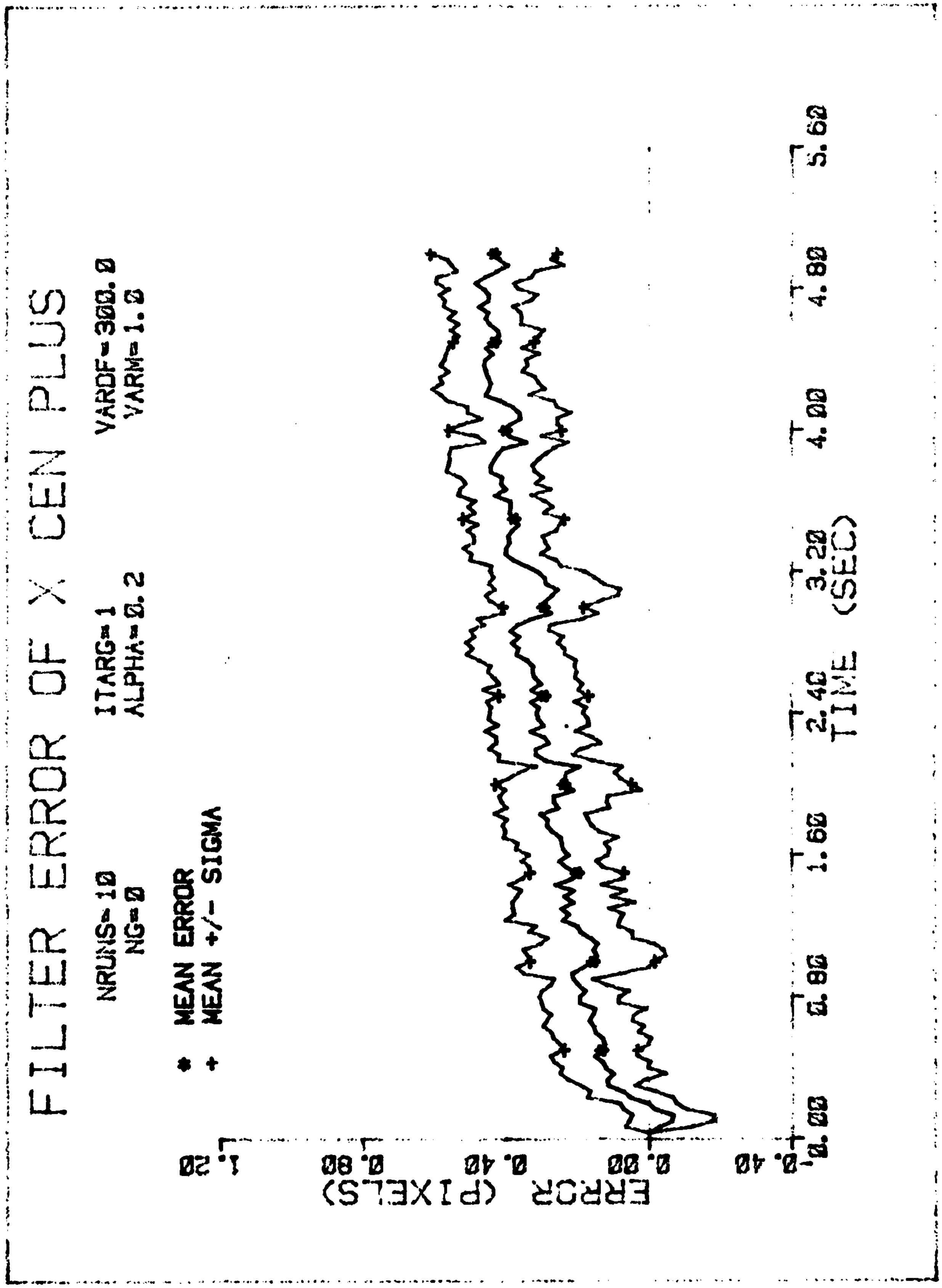


Figure E-91 Case 21 CTR Performance Plot

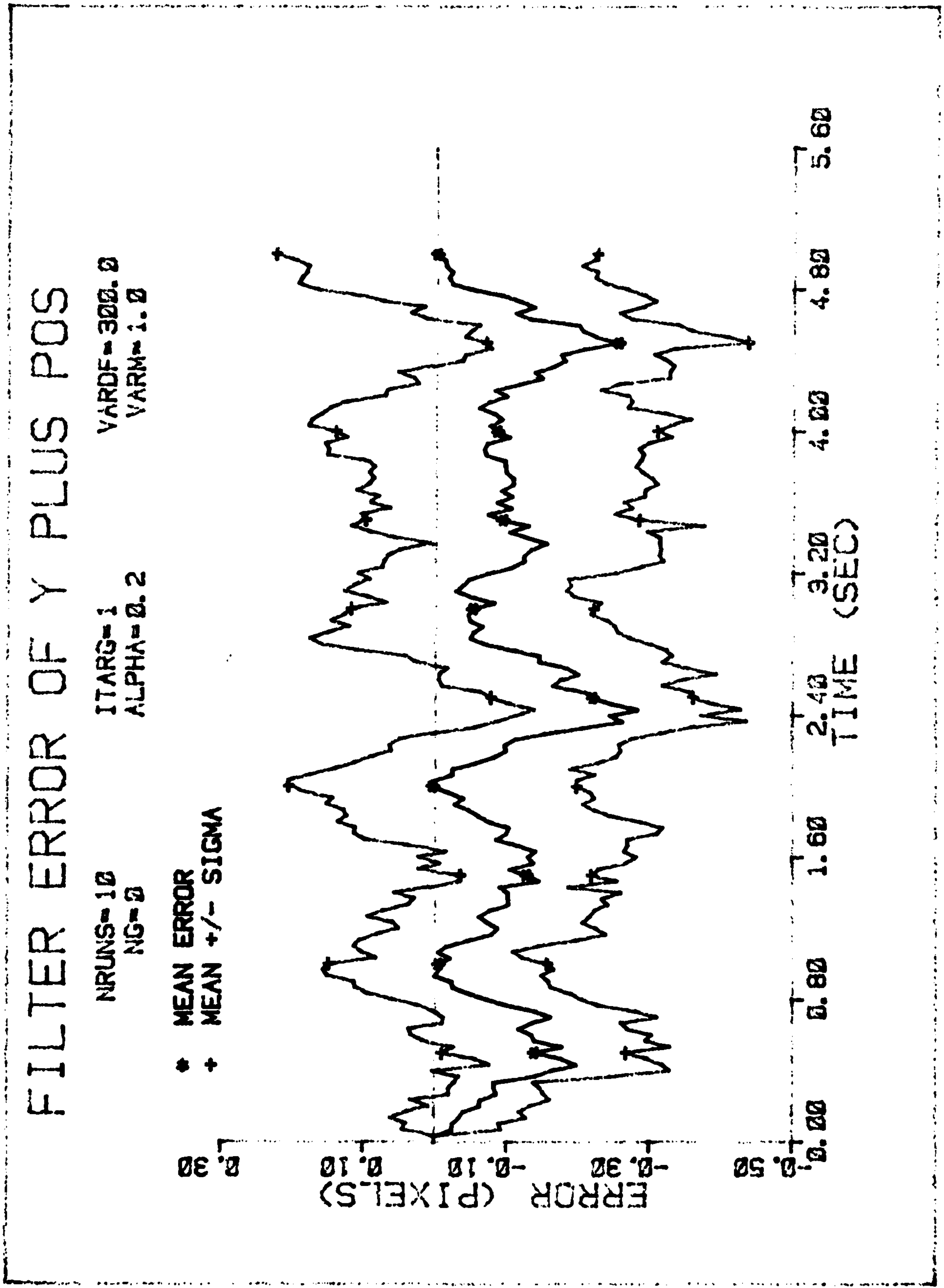


Figure E-92 Case 21 CTR Performance Plot

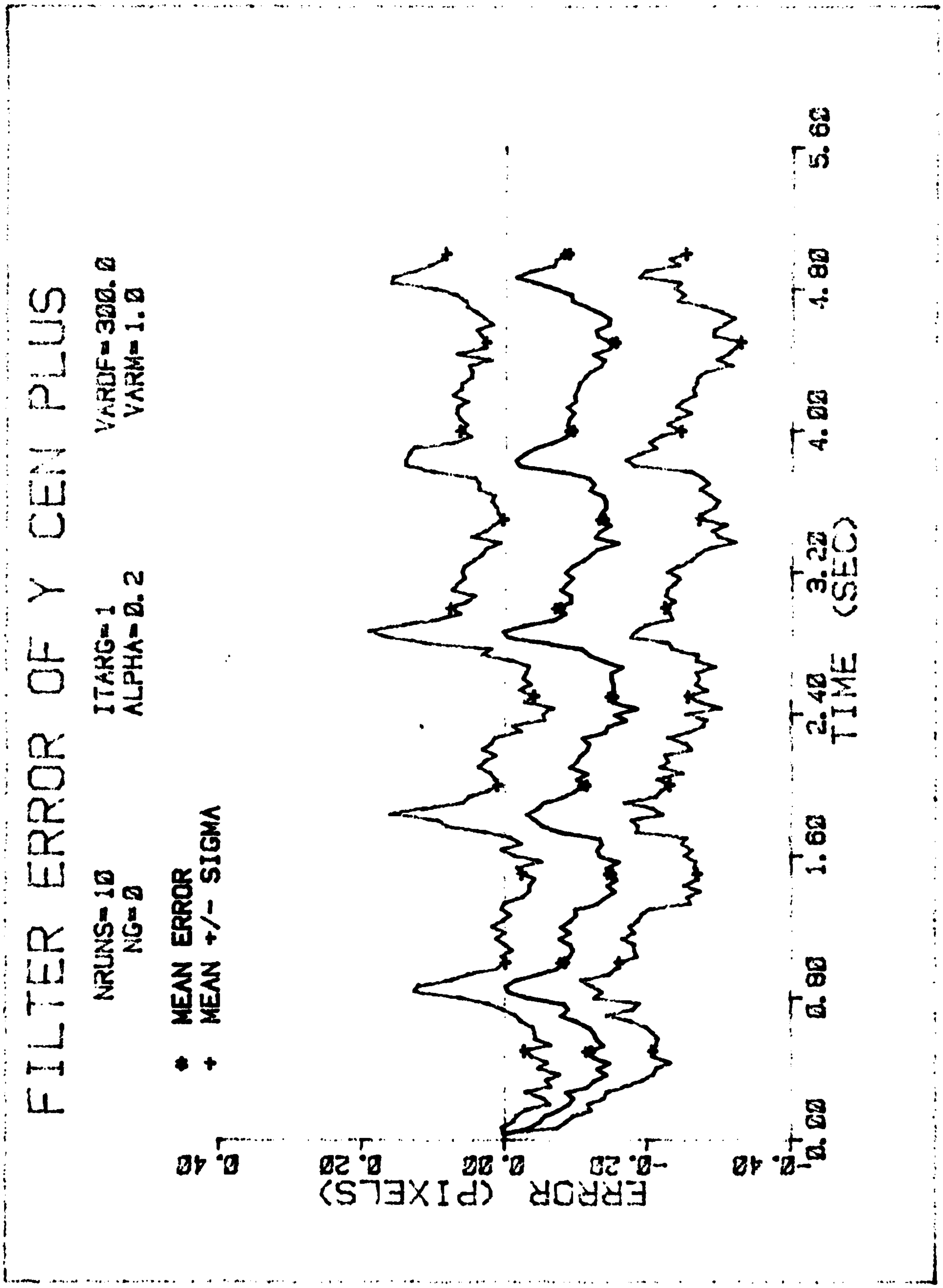


Figure E-93 Case 21 CTR Performance Plot

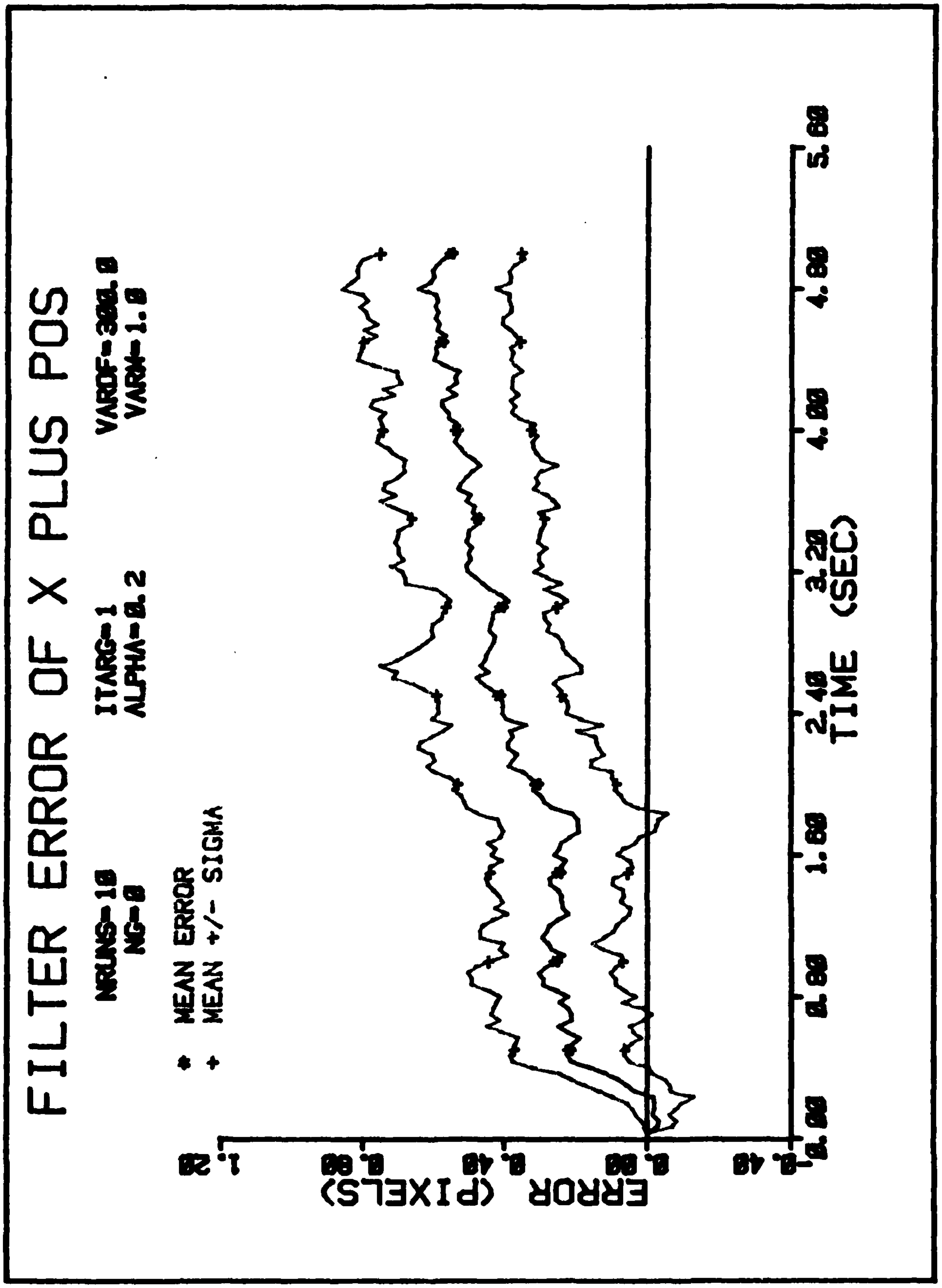


Figure E-94 Case 22 CTR Performance Plot

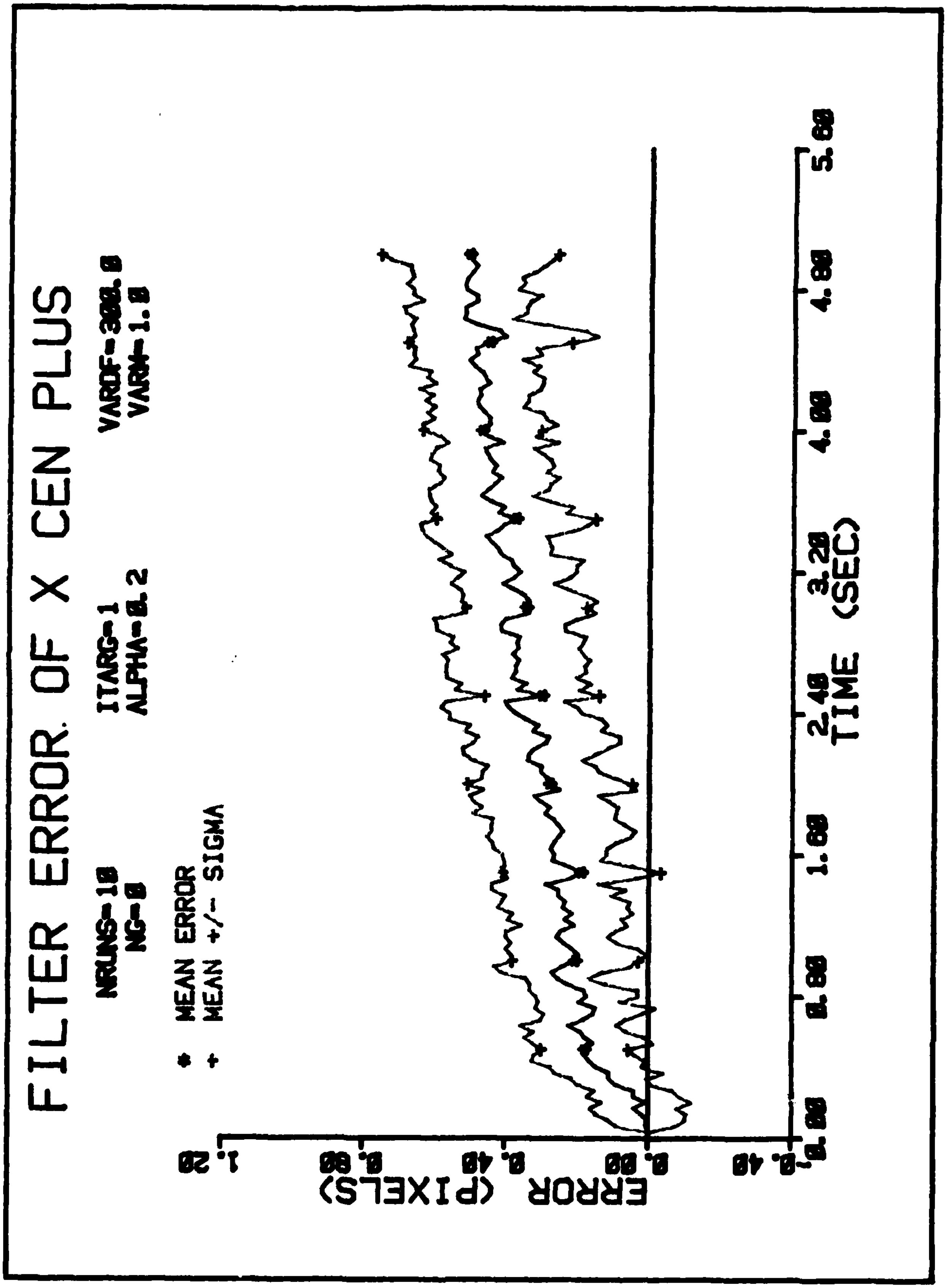


Figure E-95 Case 22 CTR Performance Plot

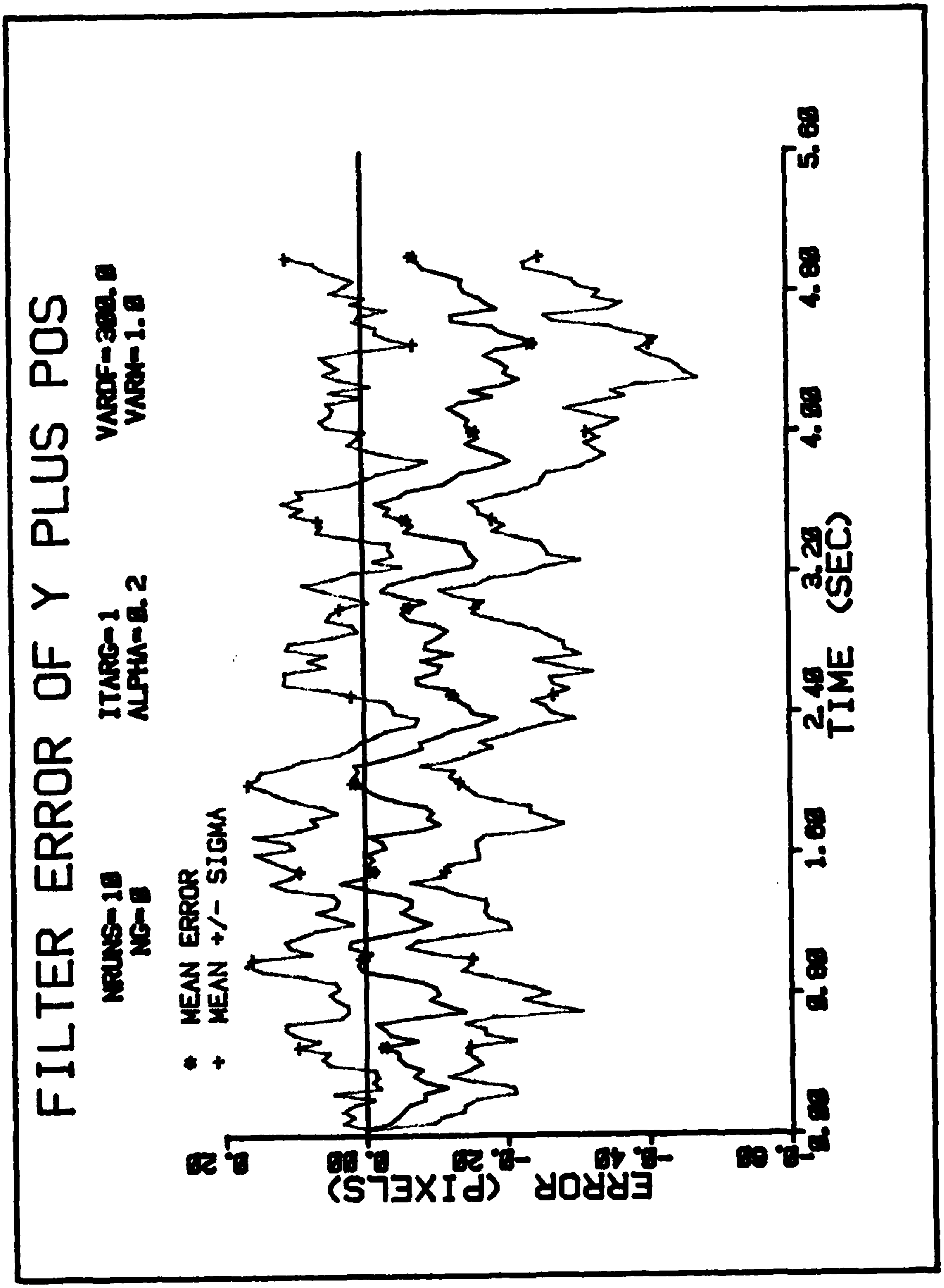


Figure E-96 Case 22 CTR Performance Plot

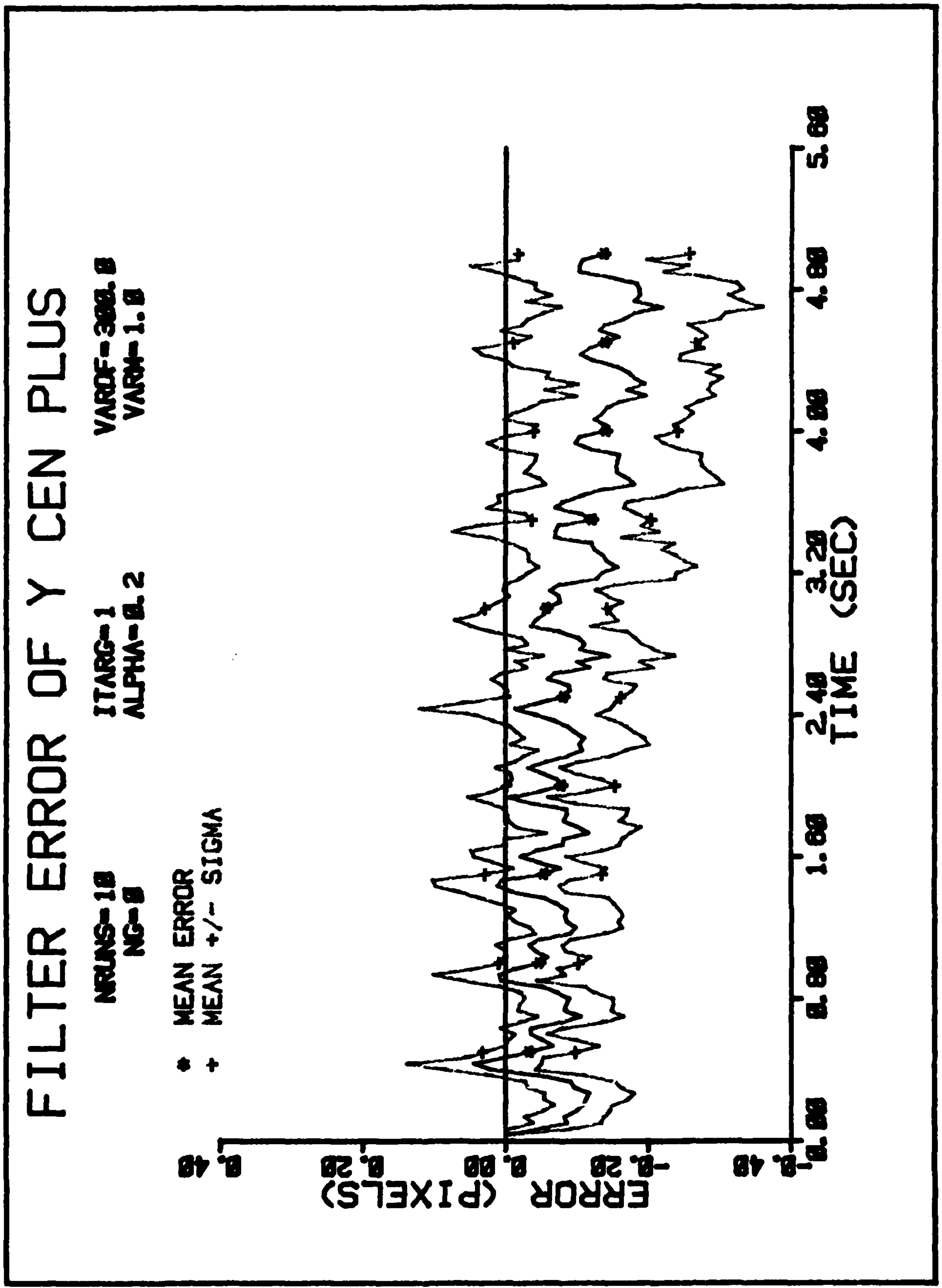


Figure E-97 Case 22 CTR Performance Plot

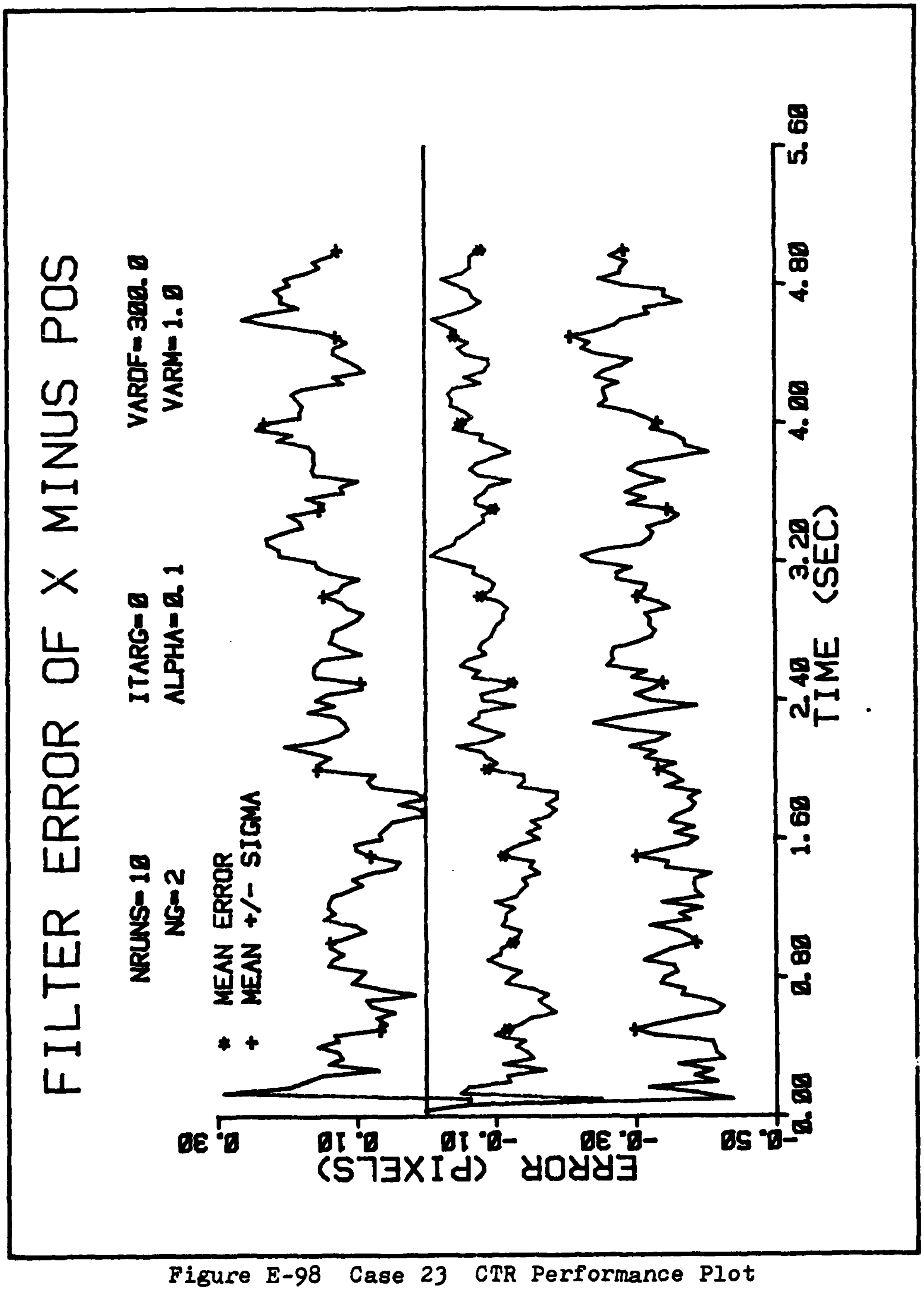
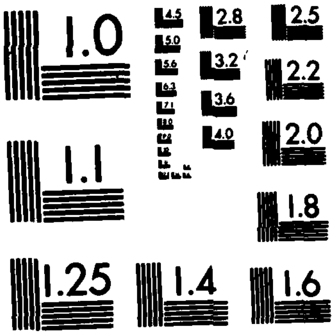


Figure E-98 Case 23 CTR Performance Plot

AD-A124 781 ENHANCED IMAGE TRACKING: ANALYSIS OF TWO ACCELERATION 5/6
MODELS IN TRACKING..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. M R KOZEMCHAK
UNCLASSIFIED DEC 82 AFIT/GEO/EE/82D-4 F/G 17/7 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

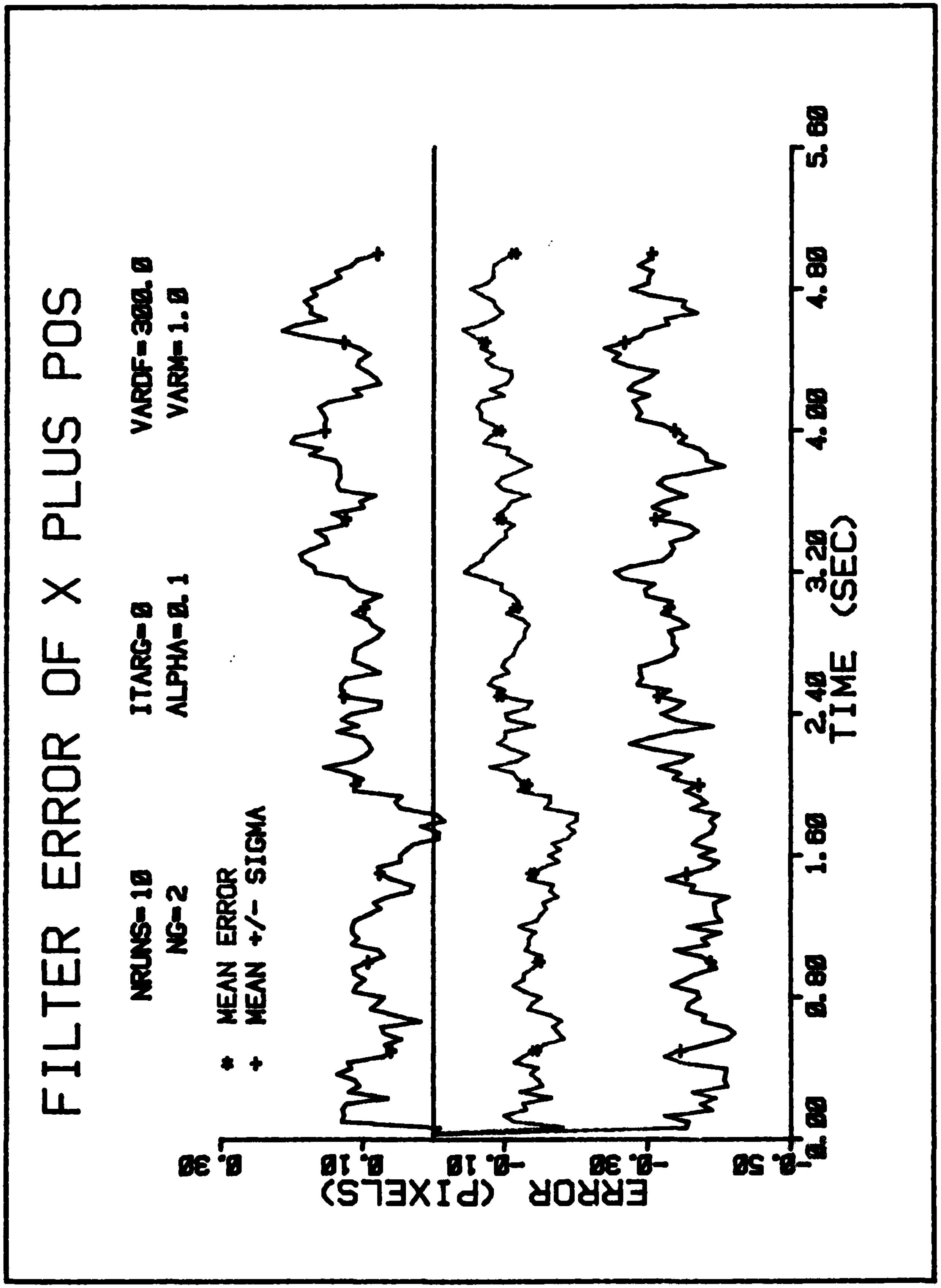


Figure E-99 Case 23 CTR Performance Plot

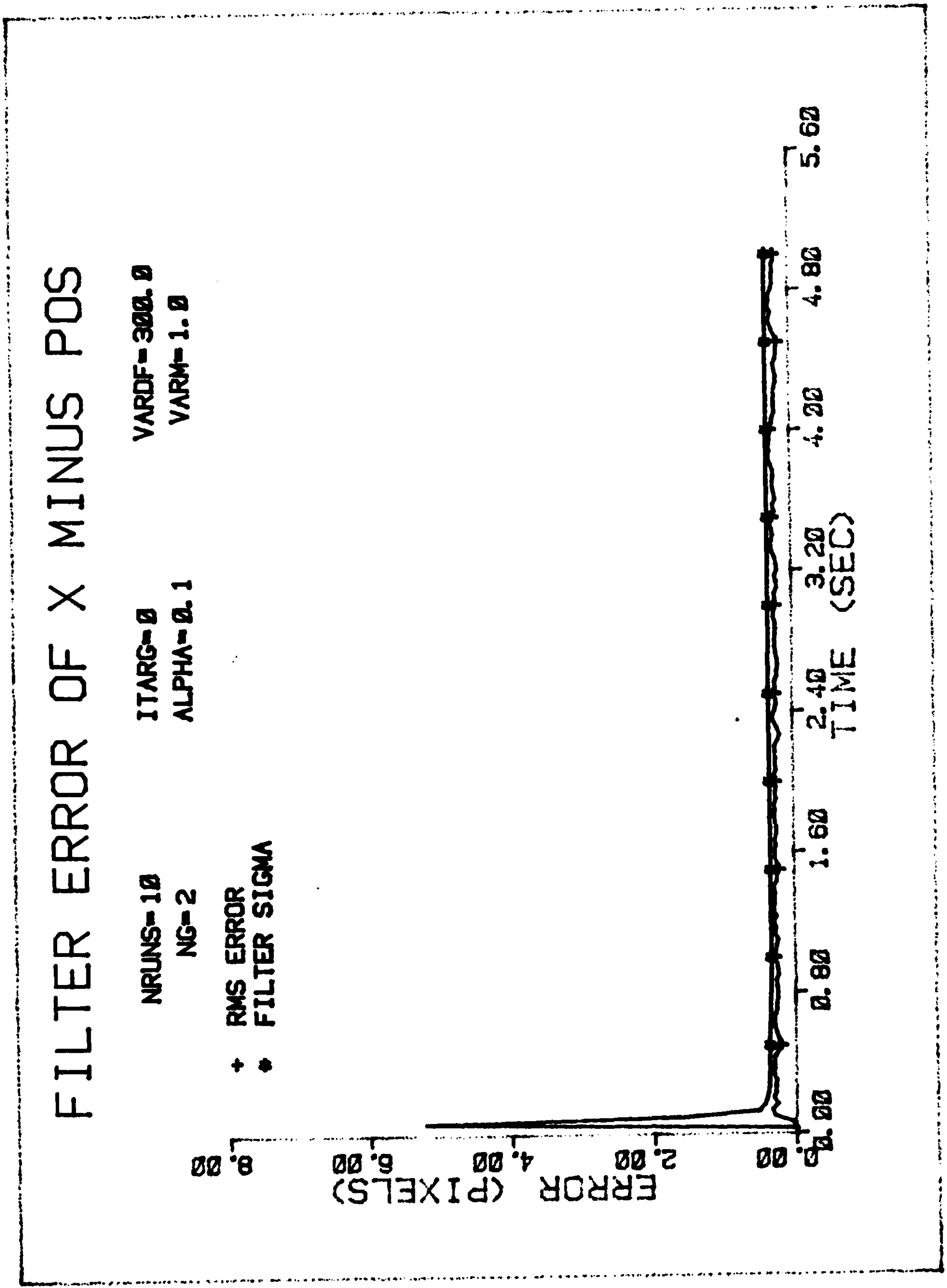


Figure E-100 Case 23 CTR Performance Plot

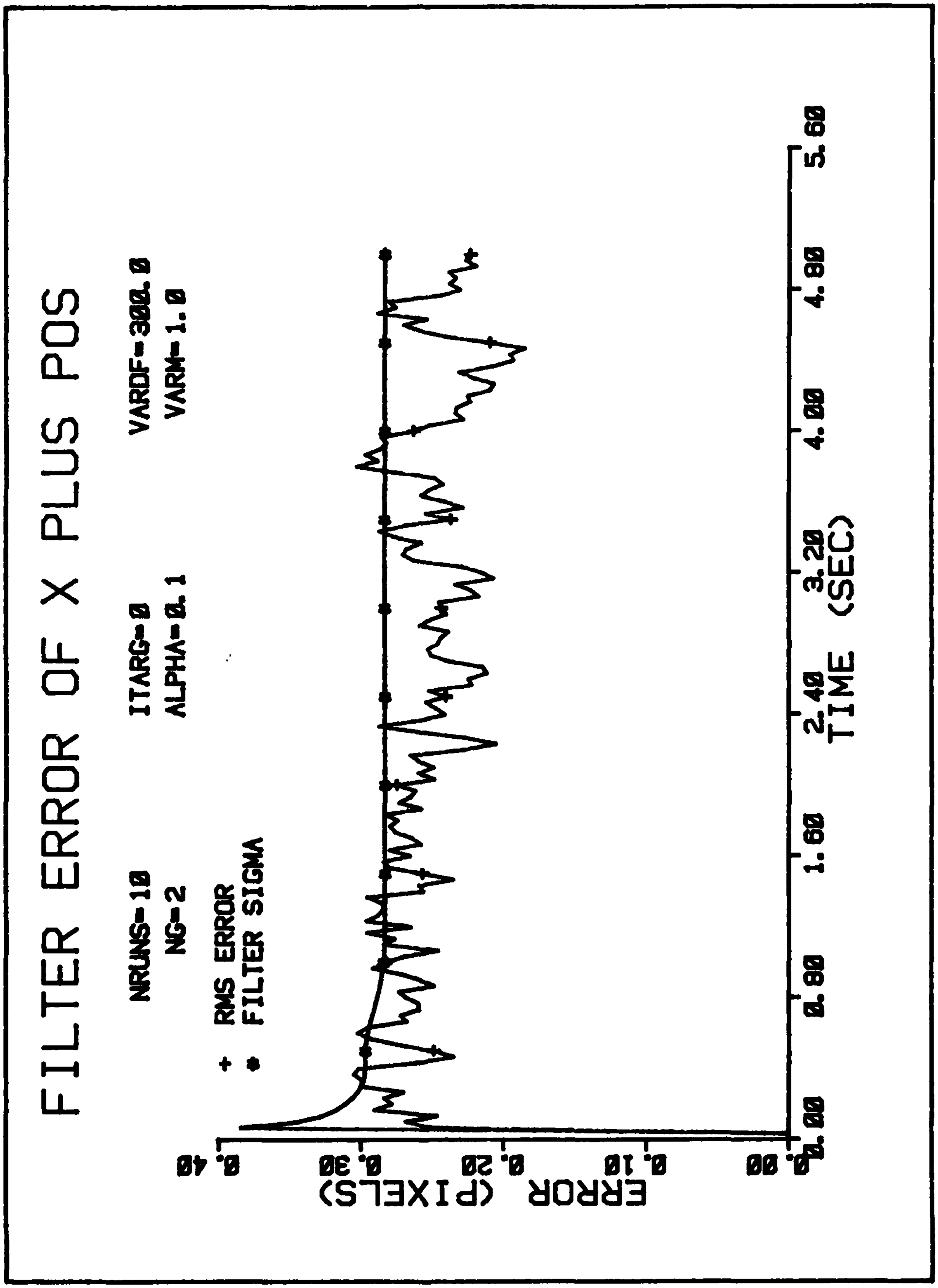


Figure E-101 Case 23 CTR Performance Plot

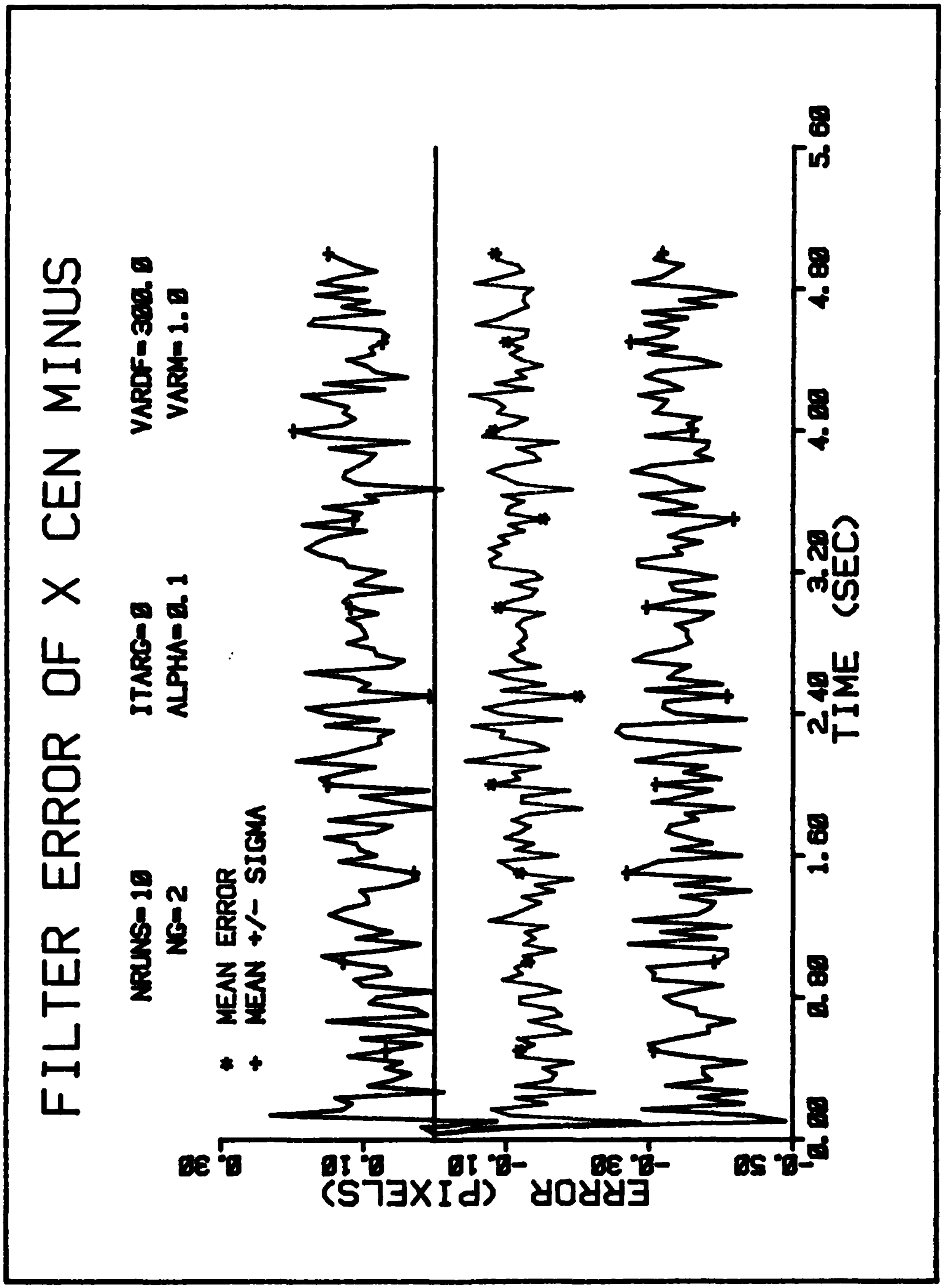


Figure E-102 Case 23 CTR Performance Plot

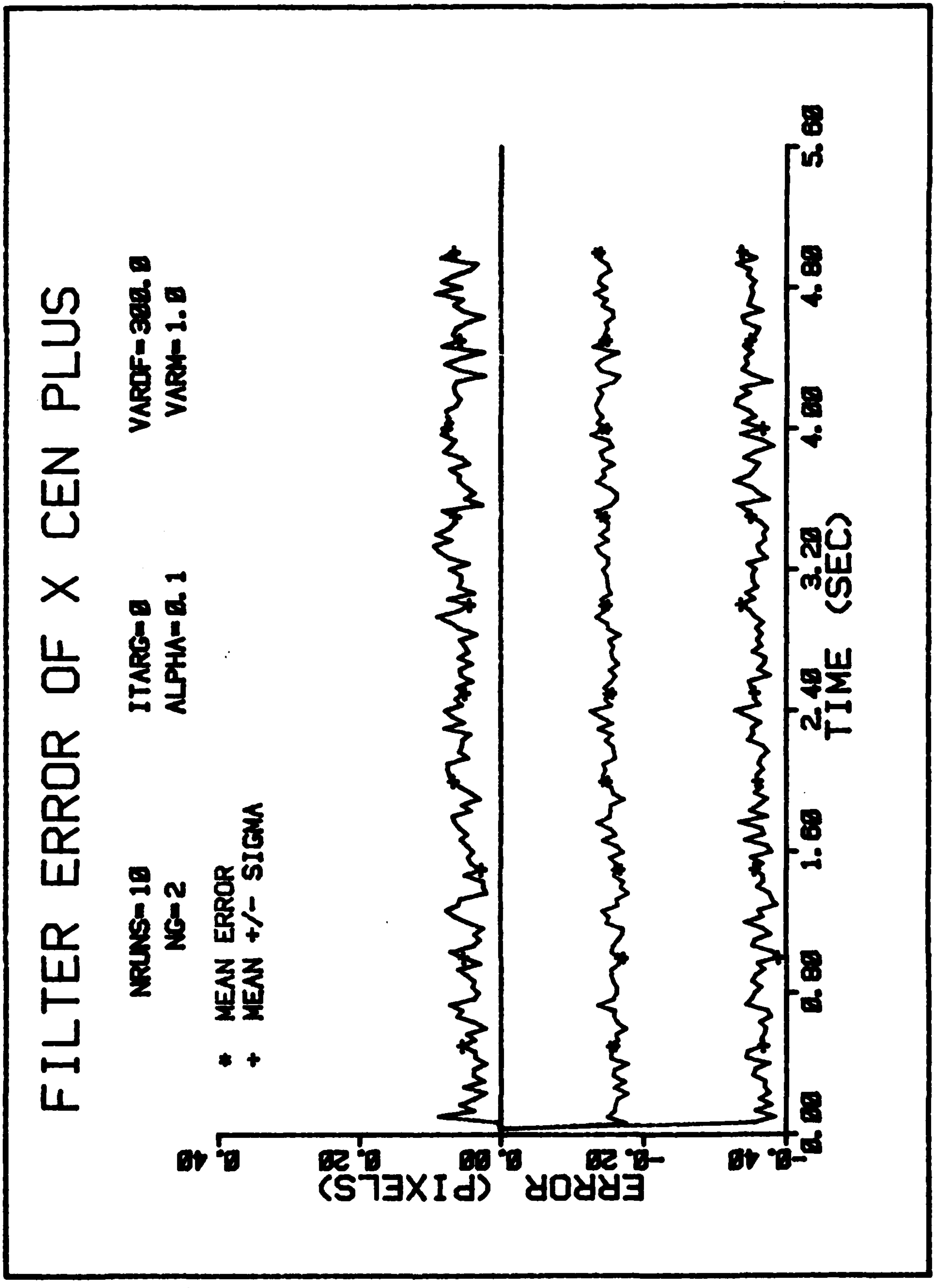


Figure E-103 Case 23 CTR Performance Plot

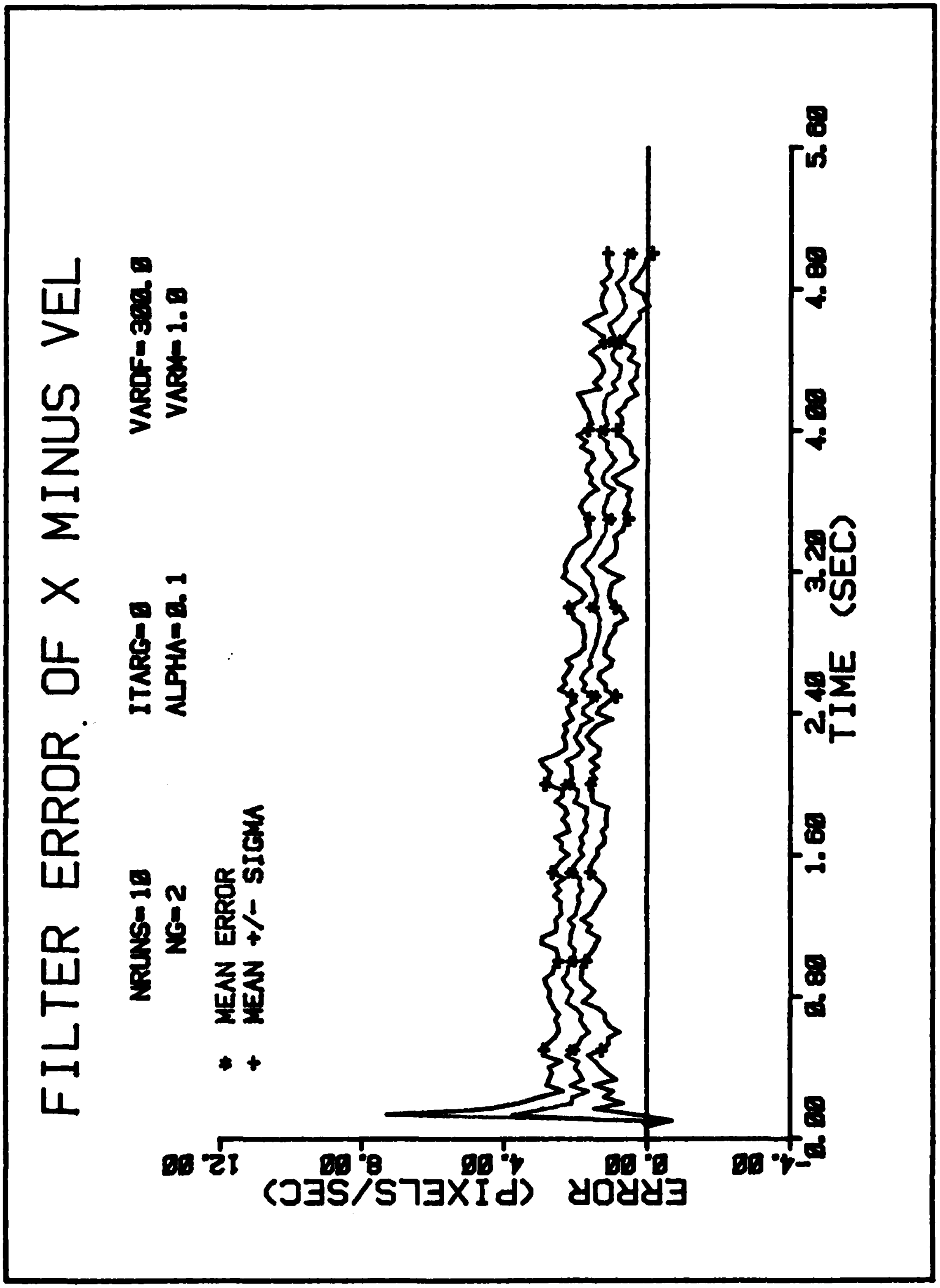


Figure E-104 Case 23 CTR Performance Plot

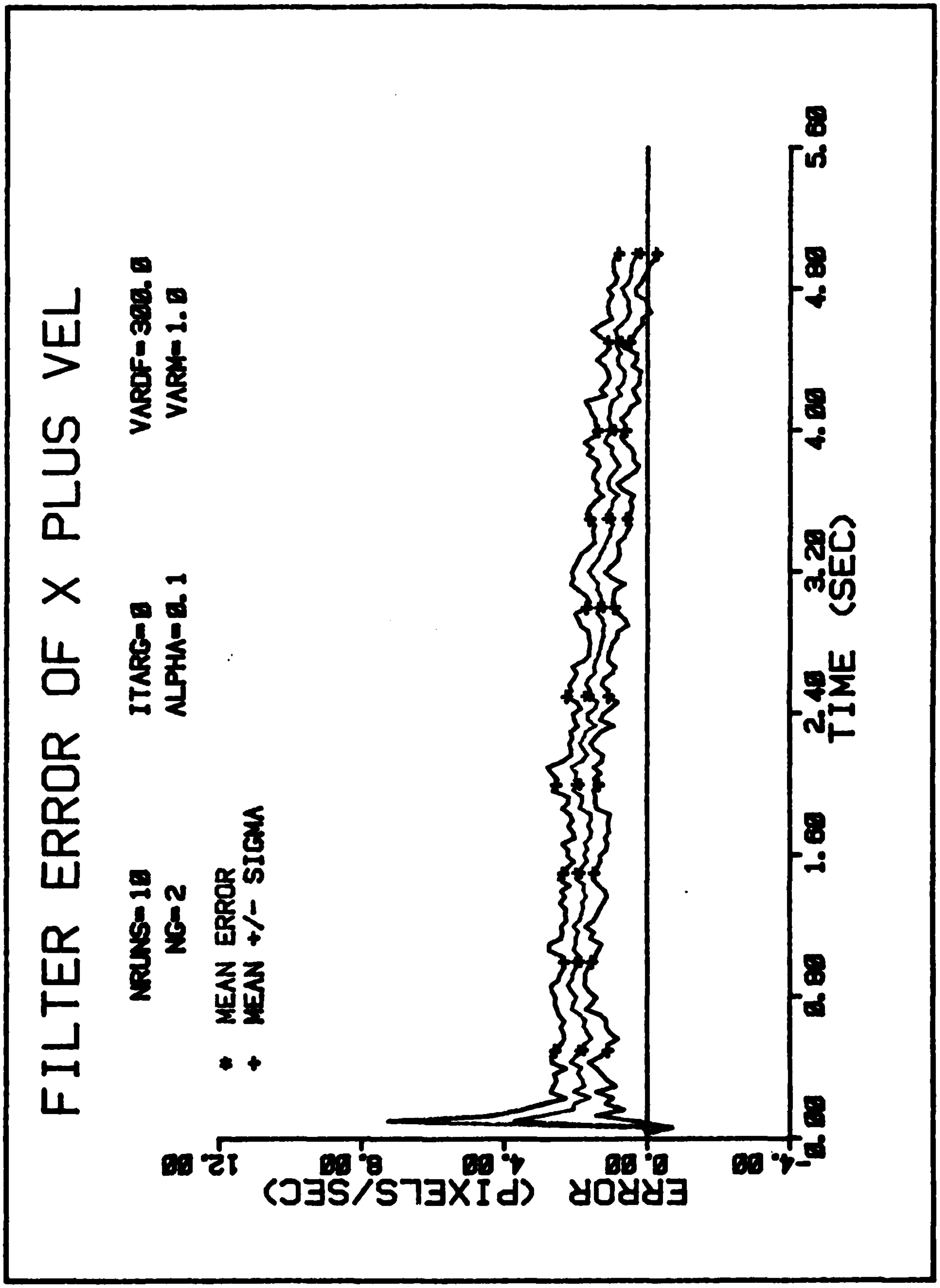


Figure E-105 Case 23 CTR Performance Plot

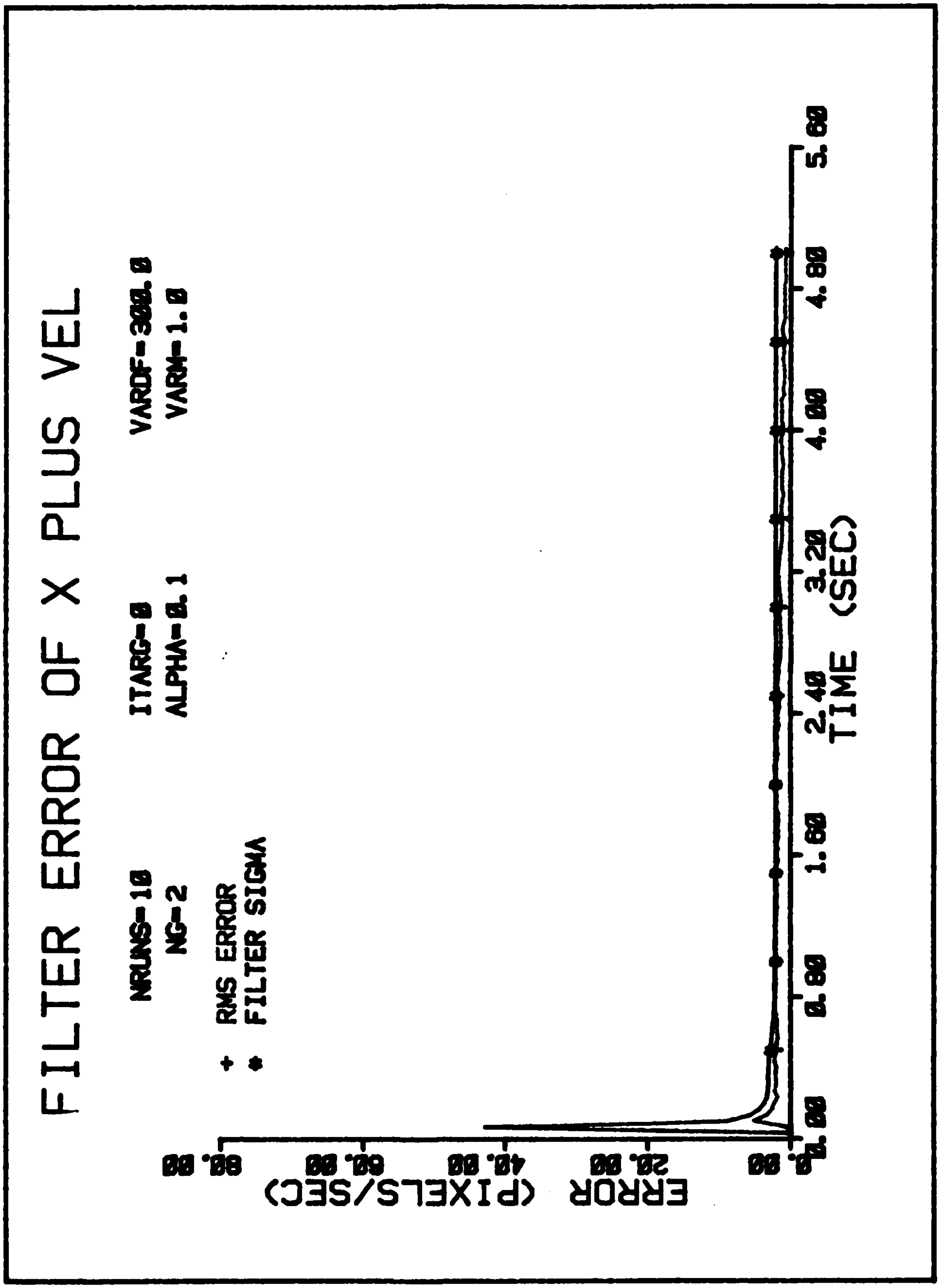


Figure E-106 Case 23 CTR Performance Plot

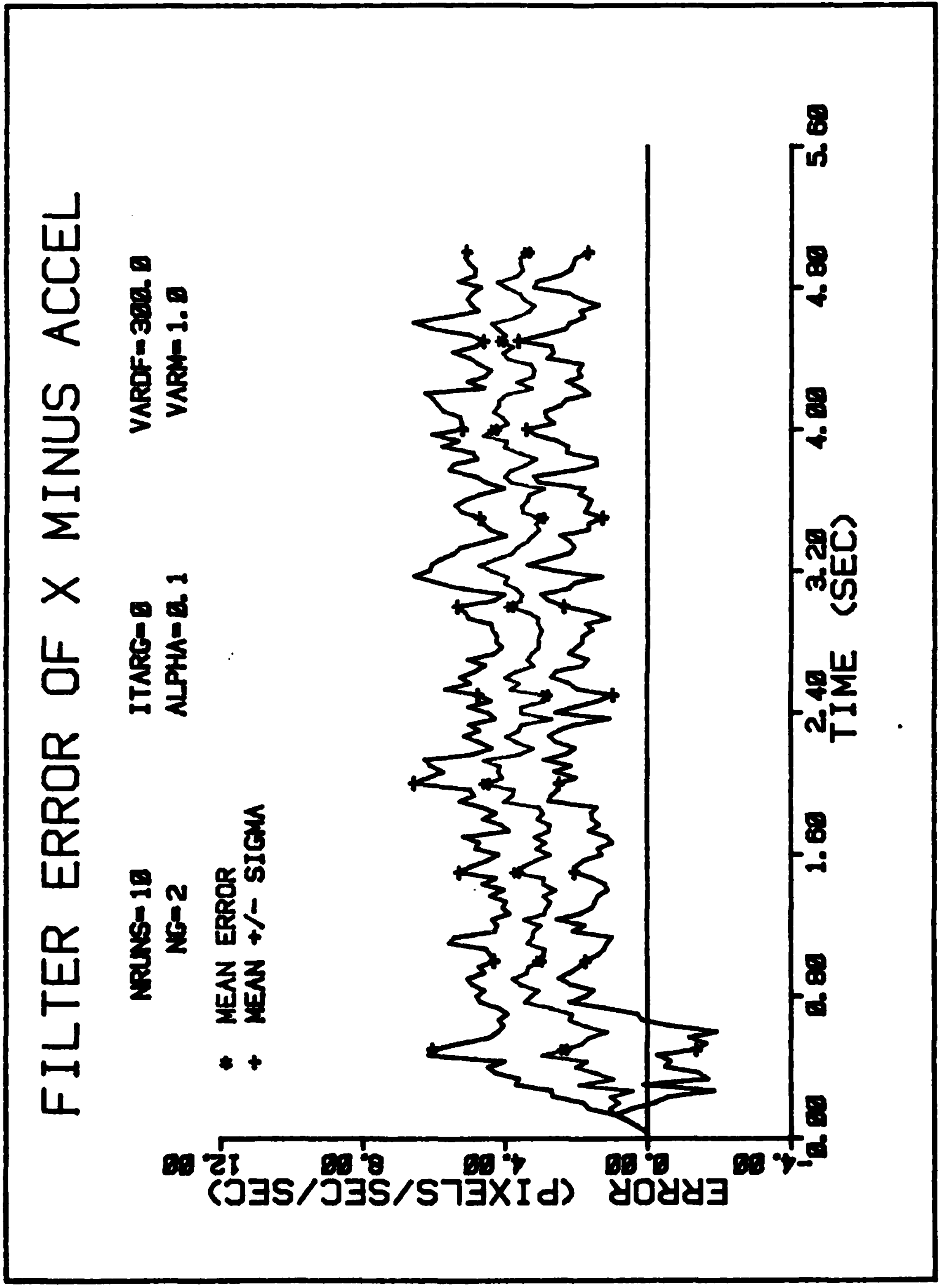


Figure E-107 Case 23 CTR Performance Plot

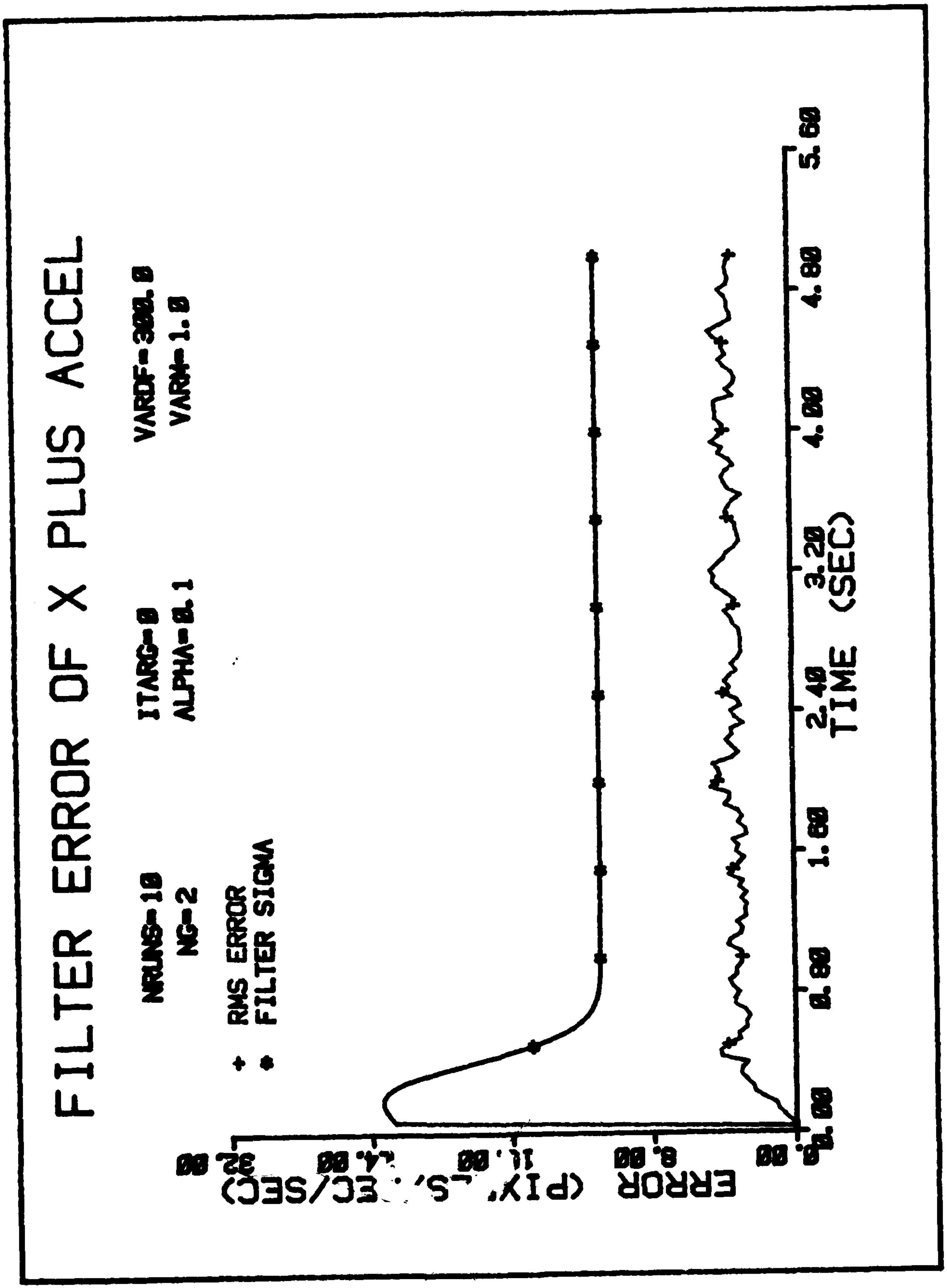


Figure E-108 Case 23 CTR Performance Plot

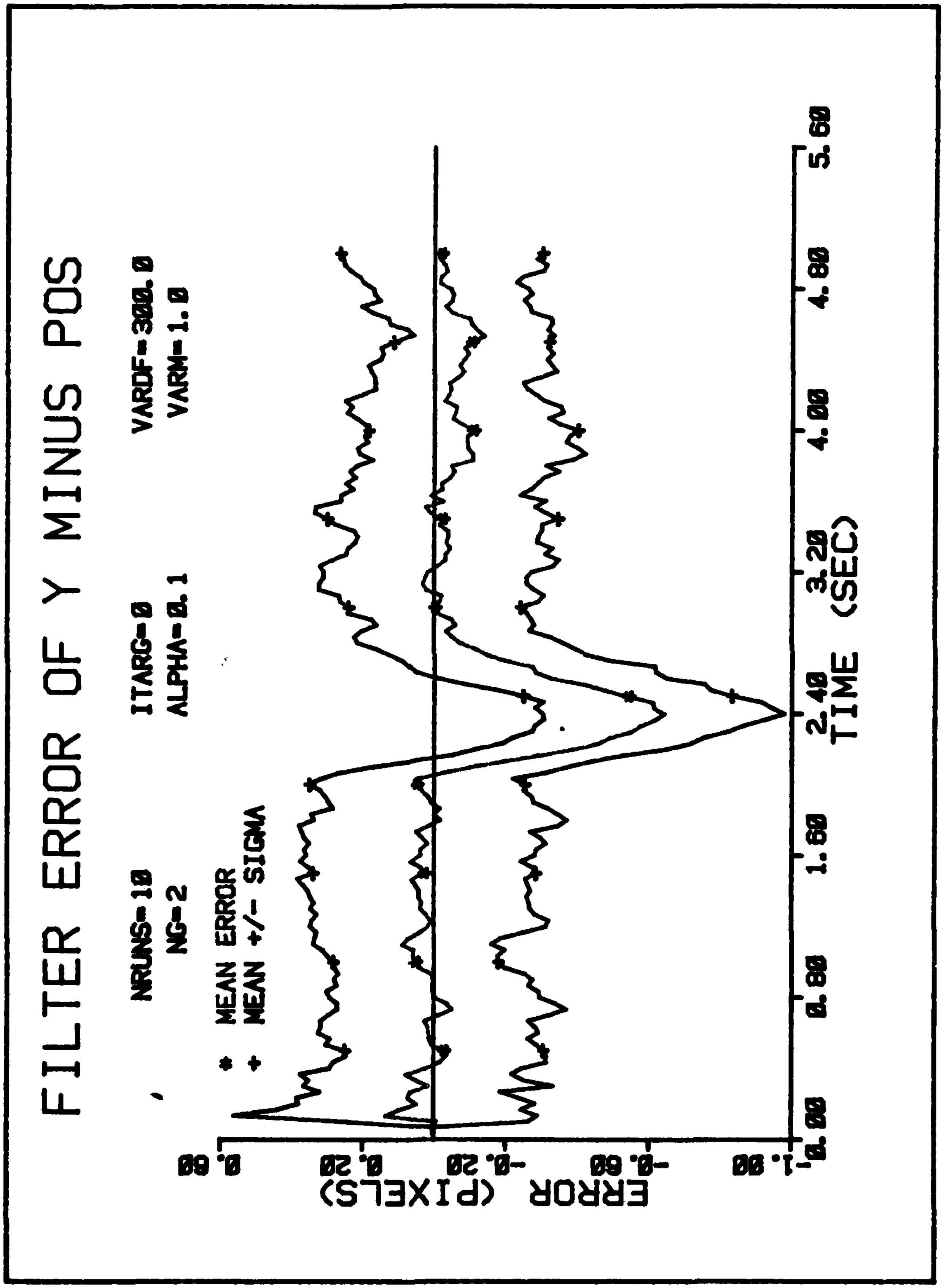


Figure E-109 Case 23 CTR Performance Plot

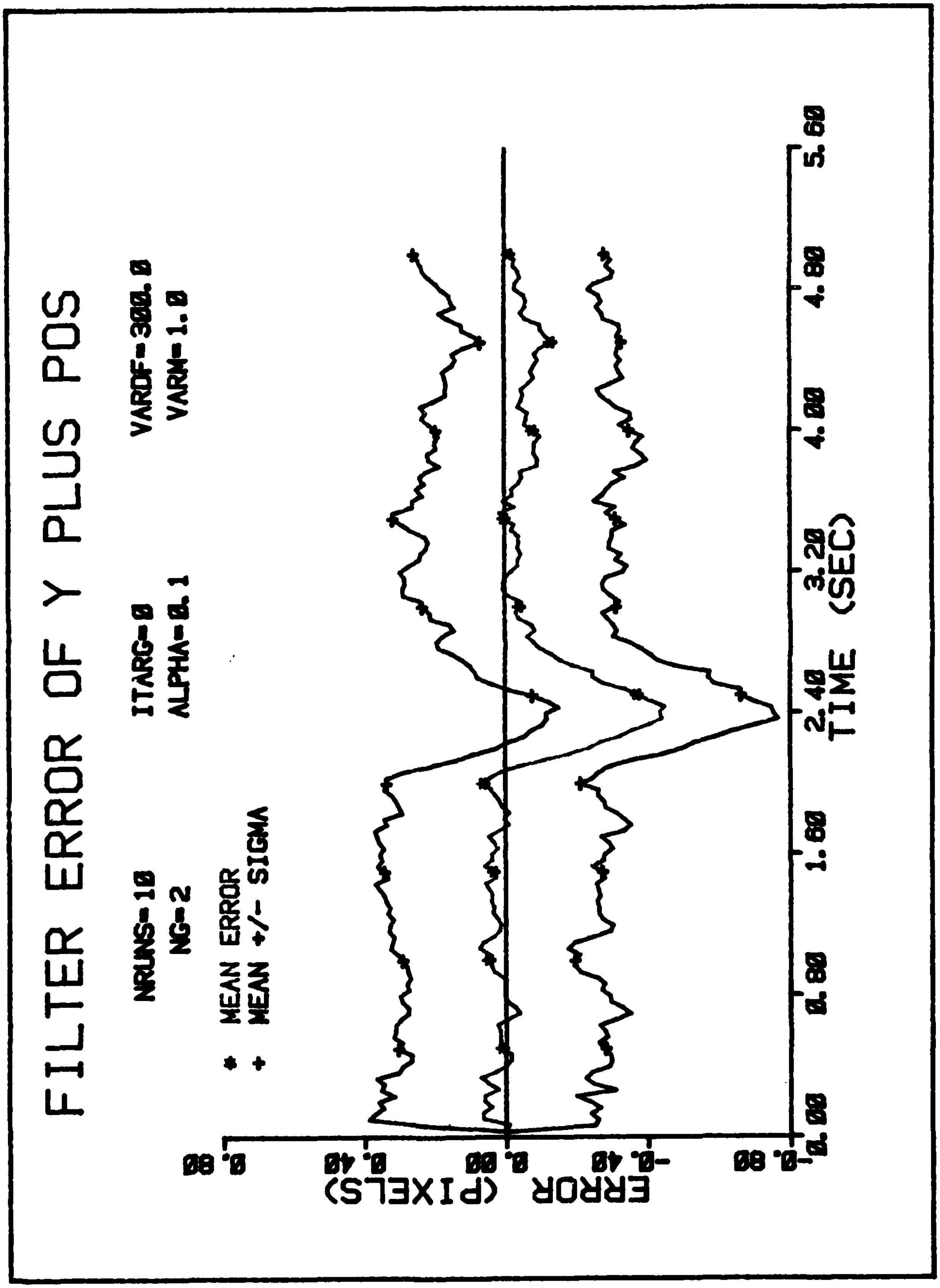


Figure E-110 Case 23 CTR Performance Plot

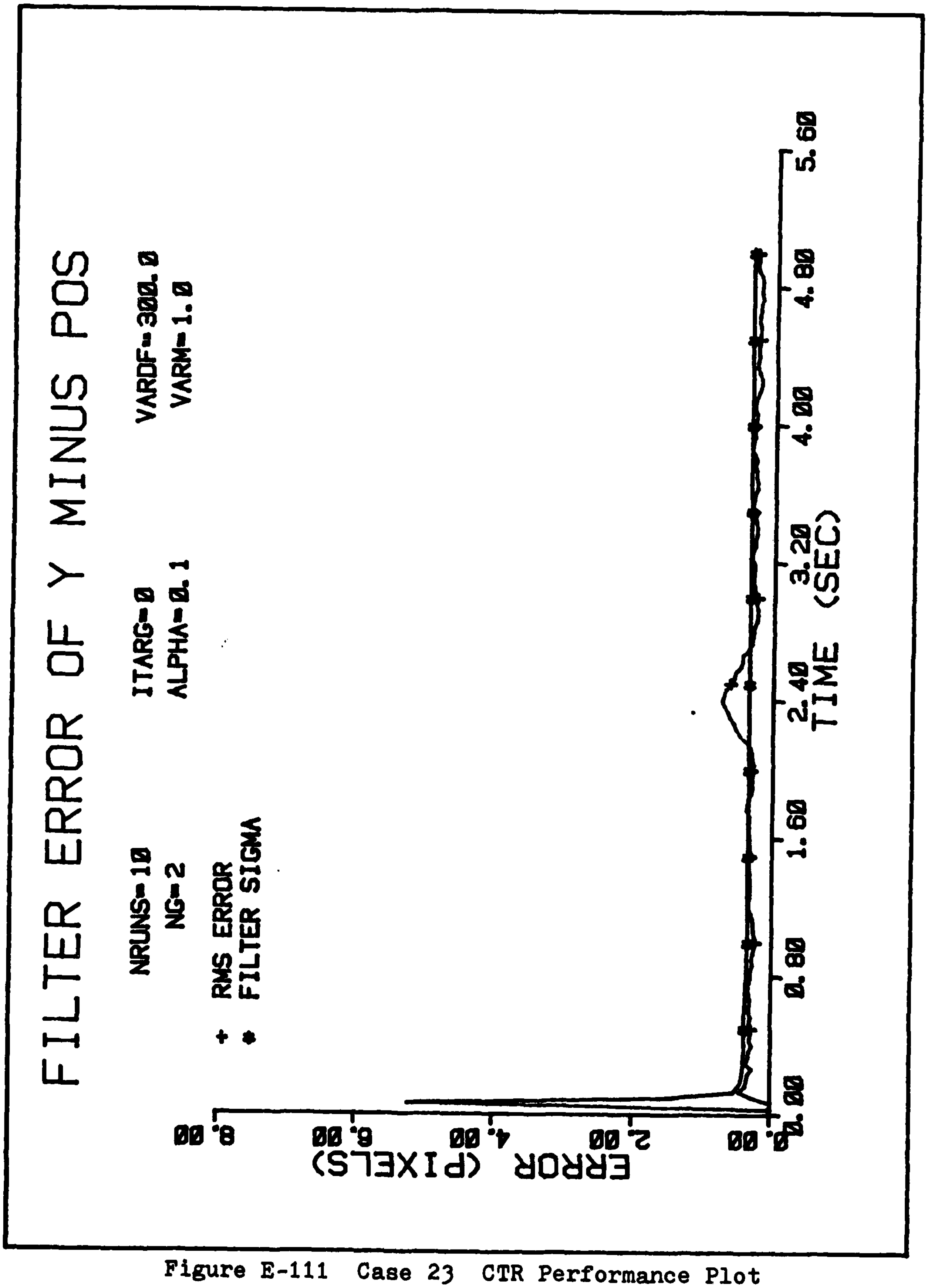


Figure E-111 Case 23 CTR Performance Plot

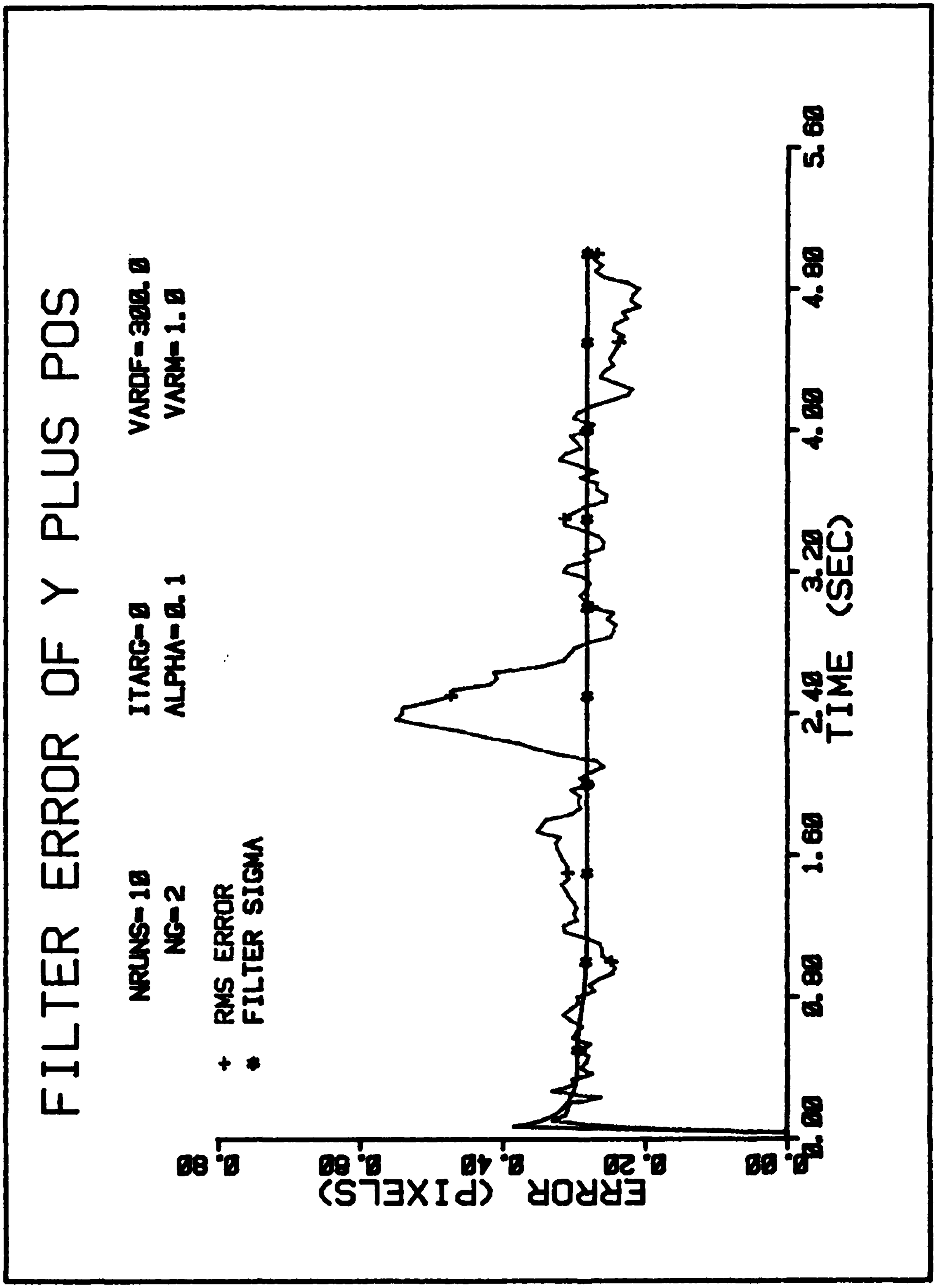


Figure E-112 Case 23 CTR Performance Plot

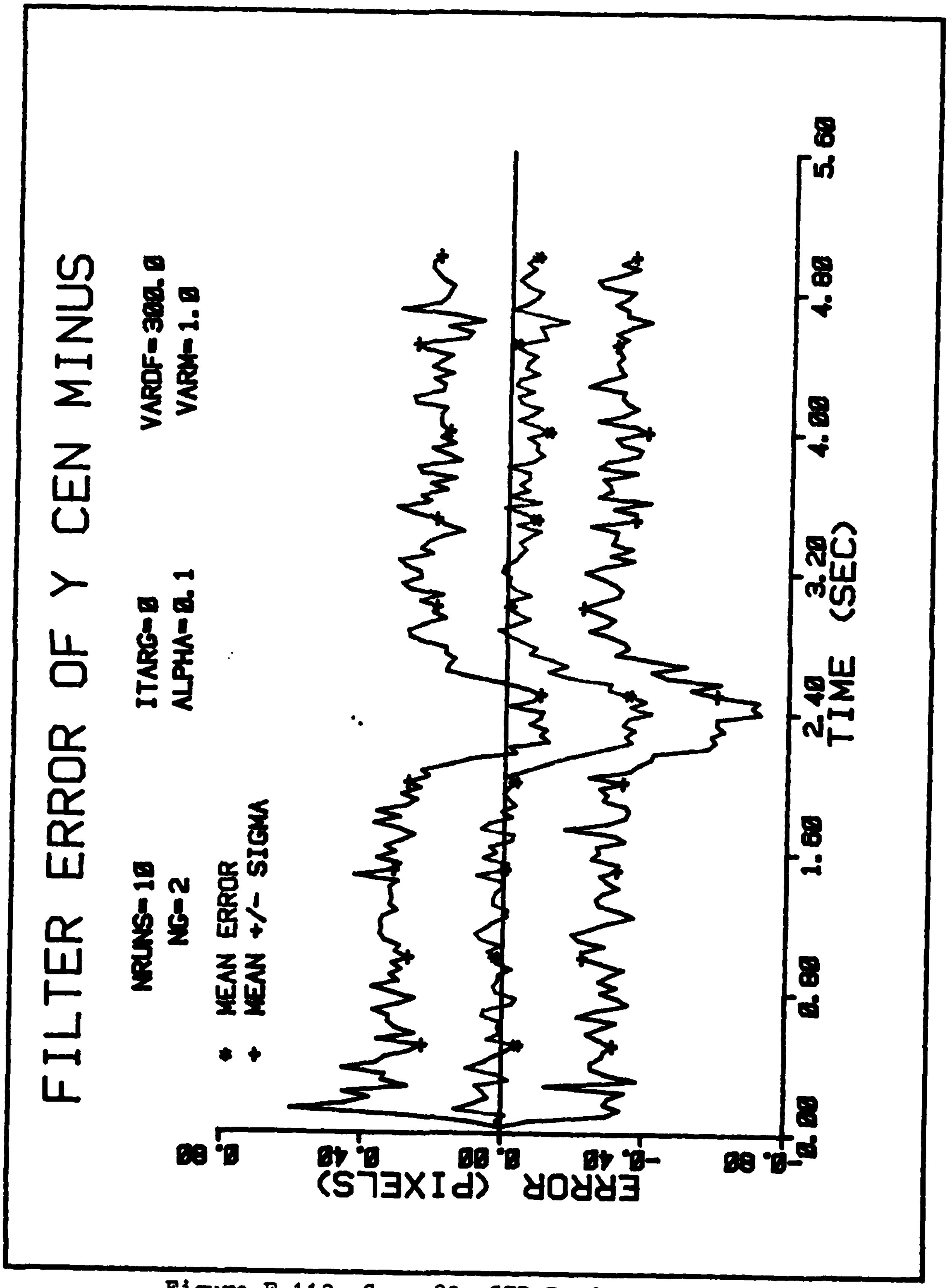


Figure E-113 Case 23 CTR Performance Plot

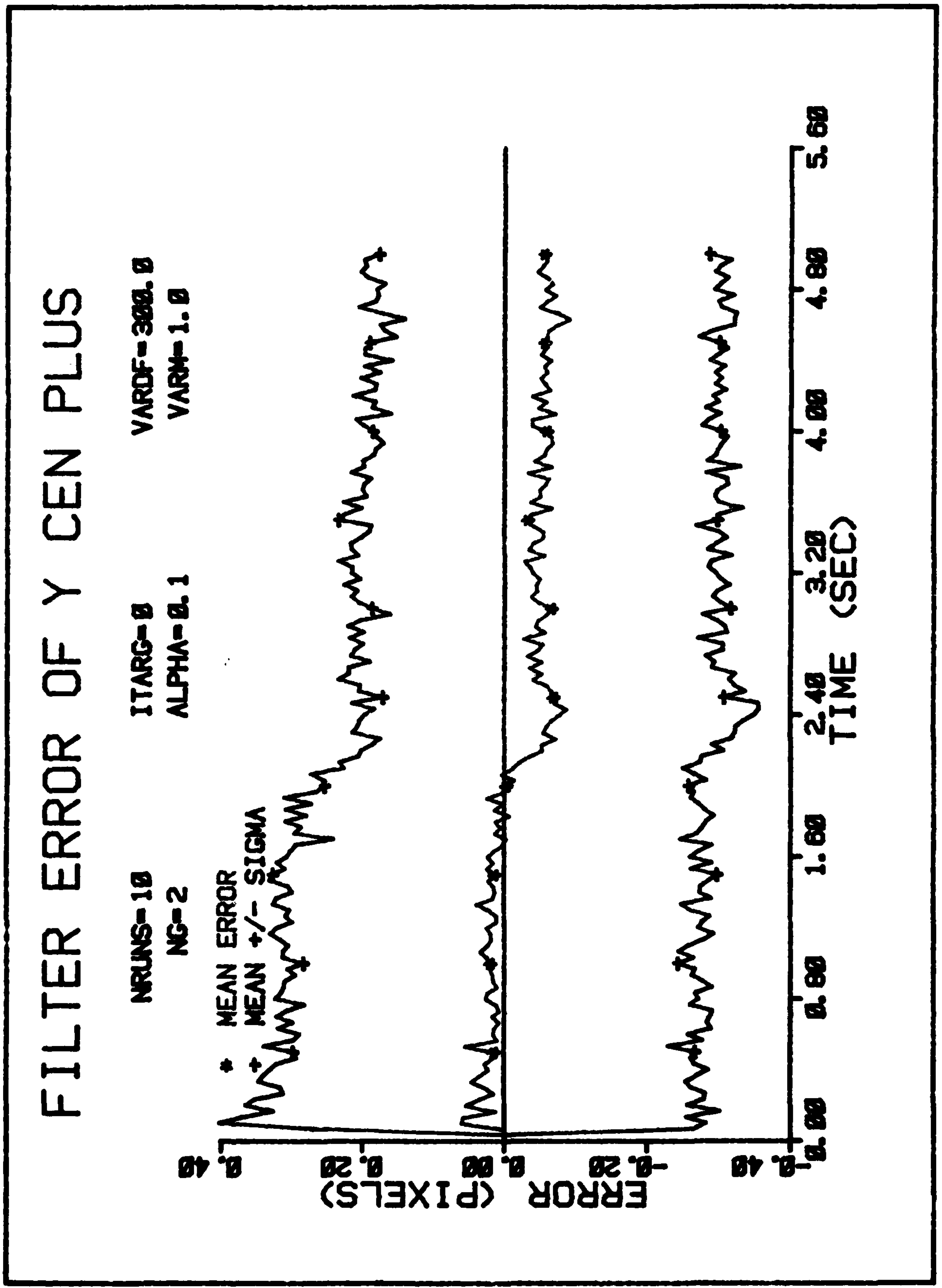


Figure E-114 Case 23 CTR Performance Plot

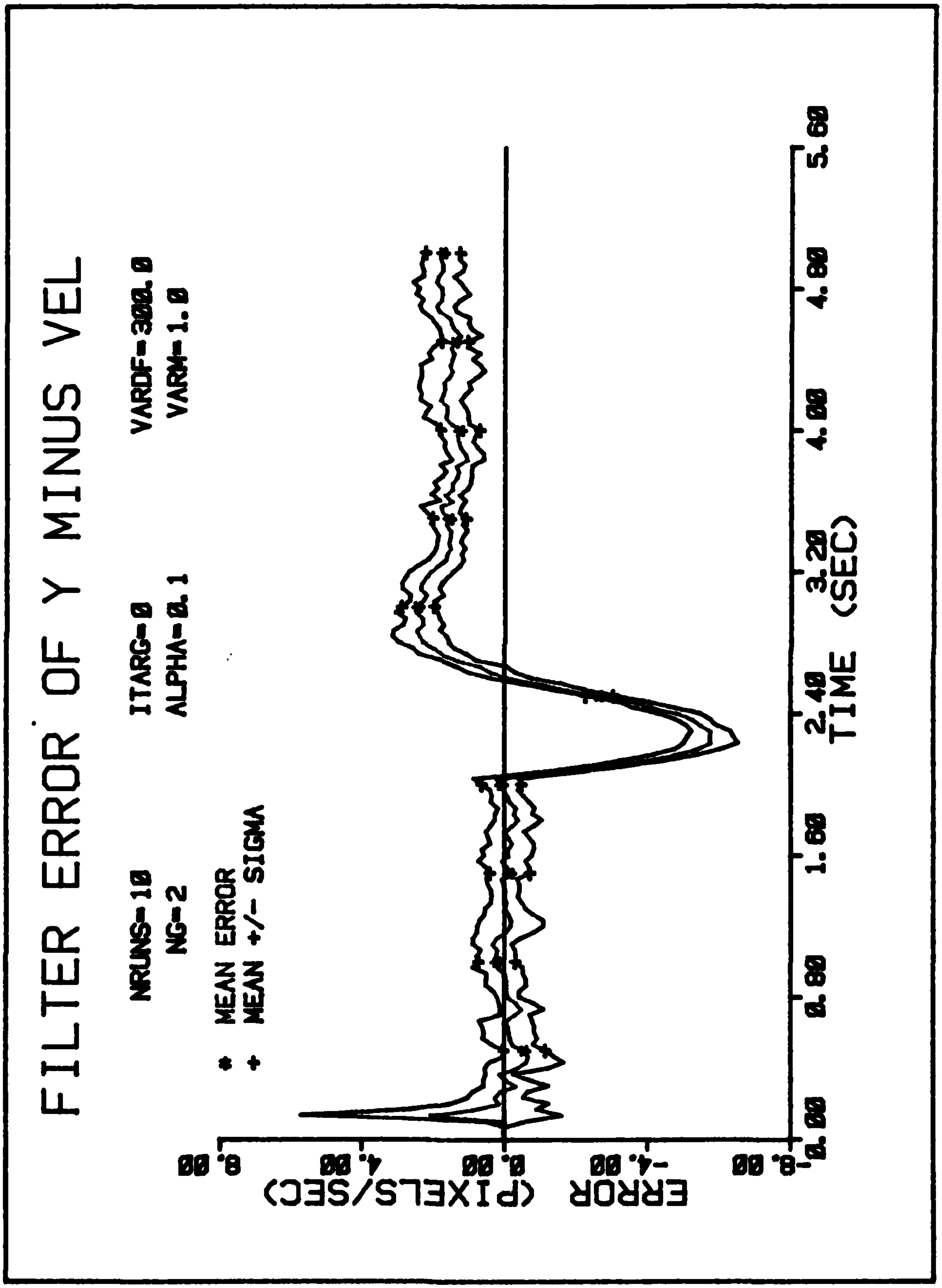


Figure E-115 Case 23 CTR Performance Plot

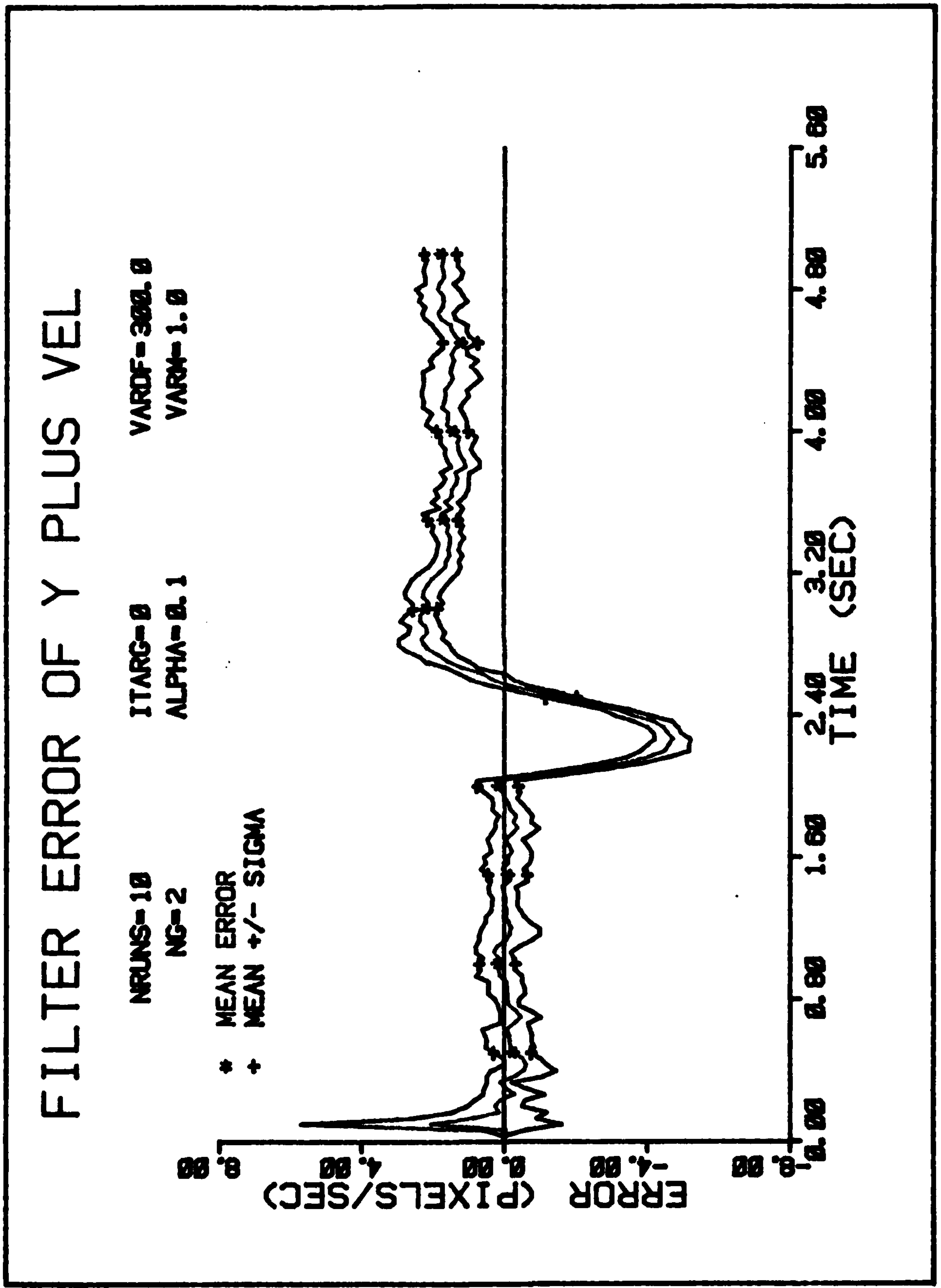


Figure E-116 Case 23 CTR Performance Plot

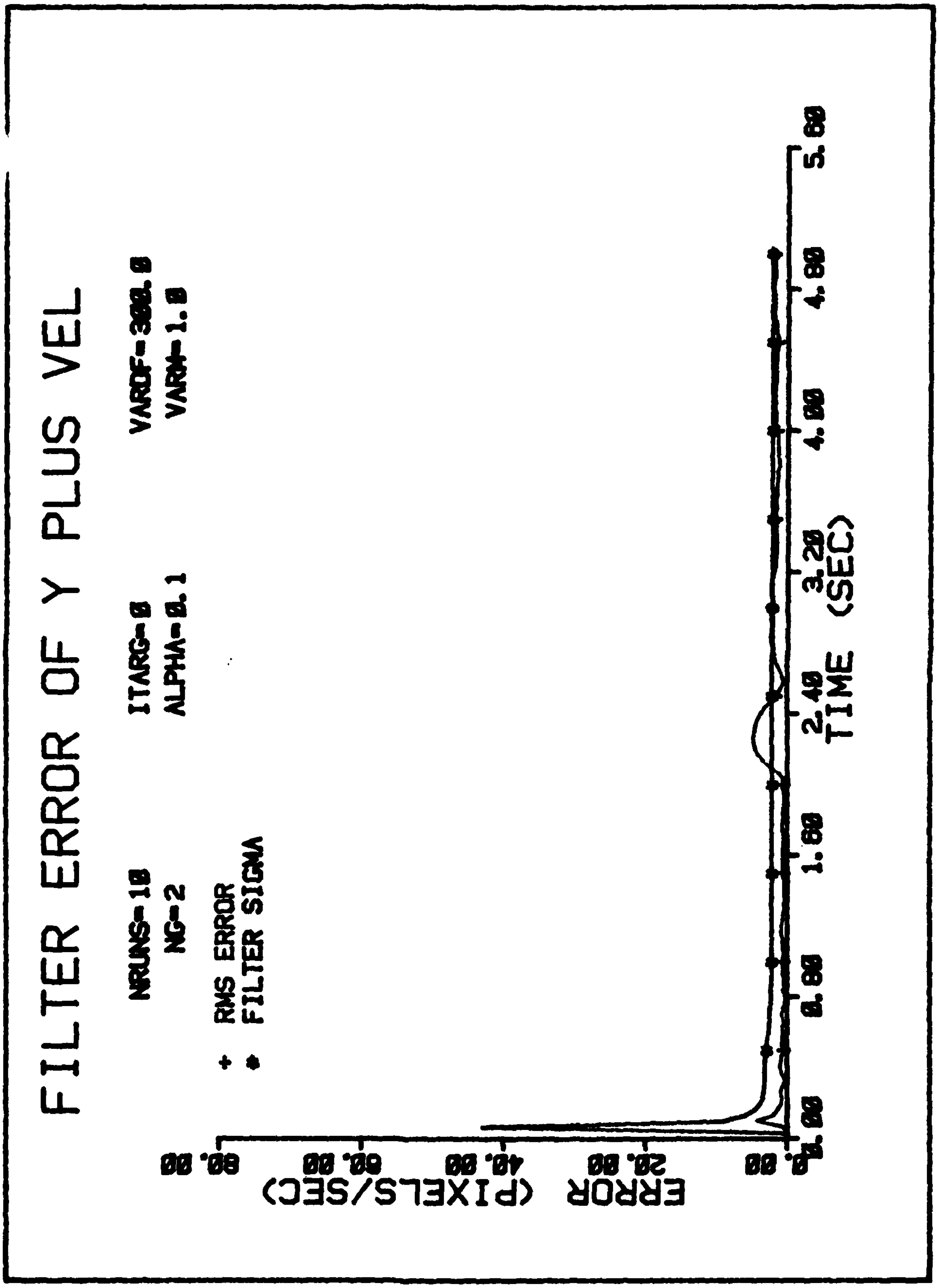


Figure E-117 Case 23 CTR Performance Plot

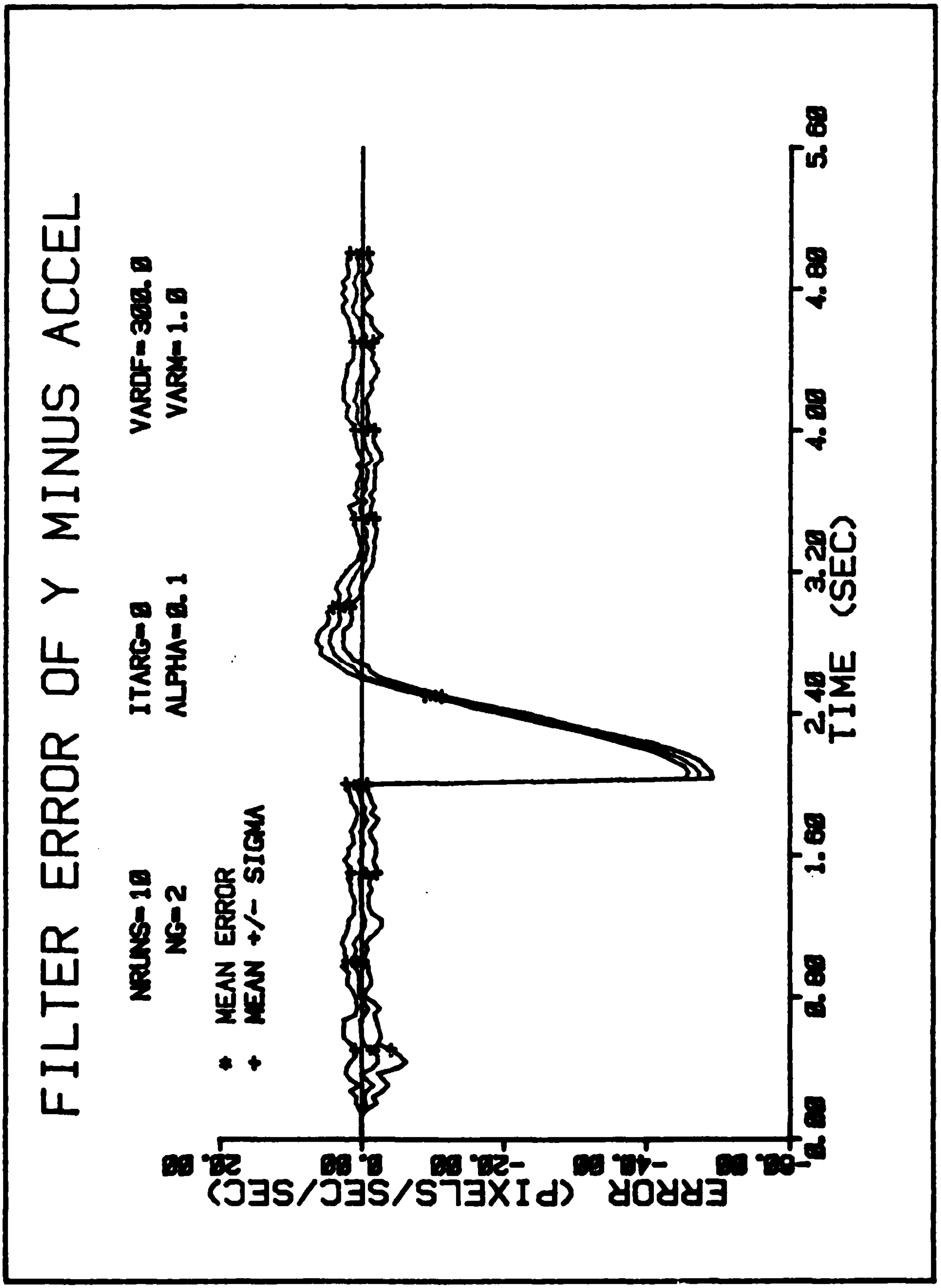


Figure E-118 Case 23 CTR Performance Plot

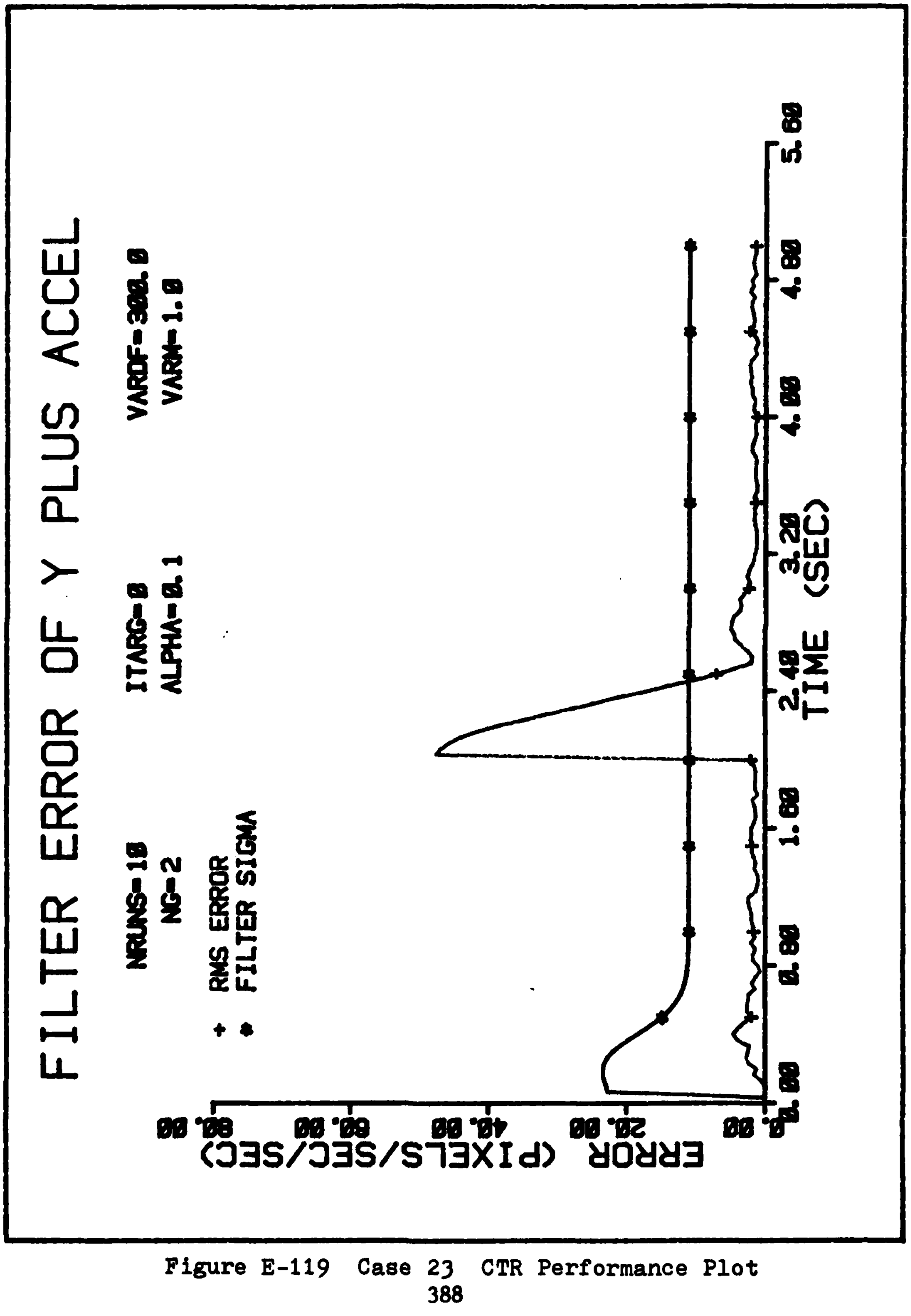


Figure E-119 Case 23 CTR Performance Plot

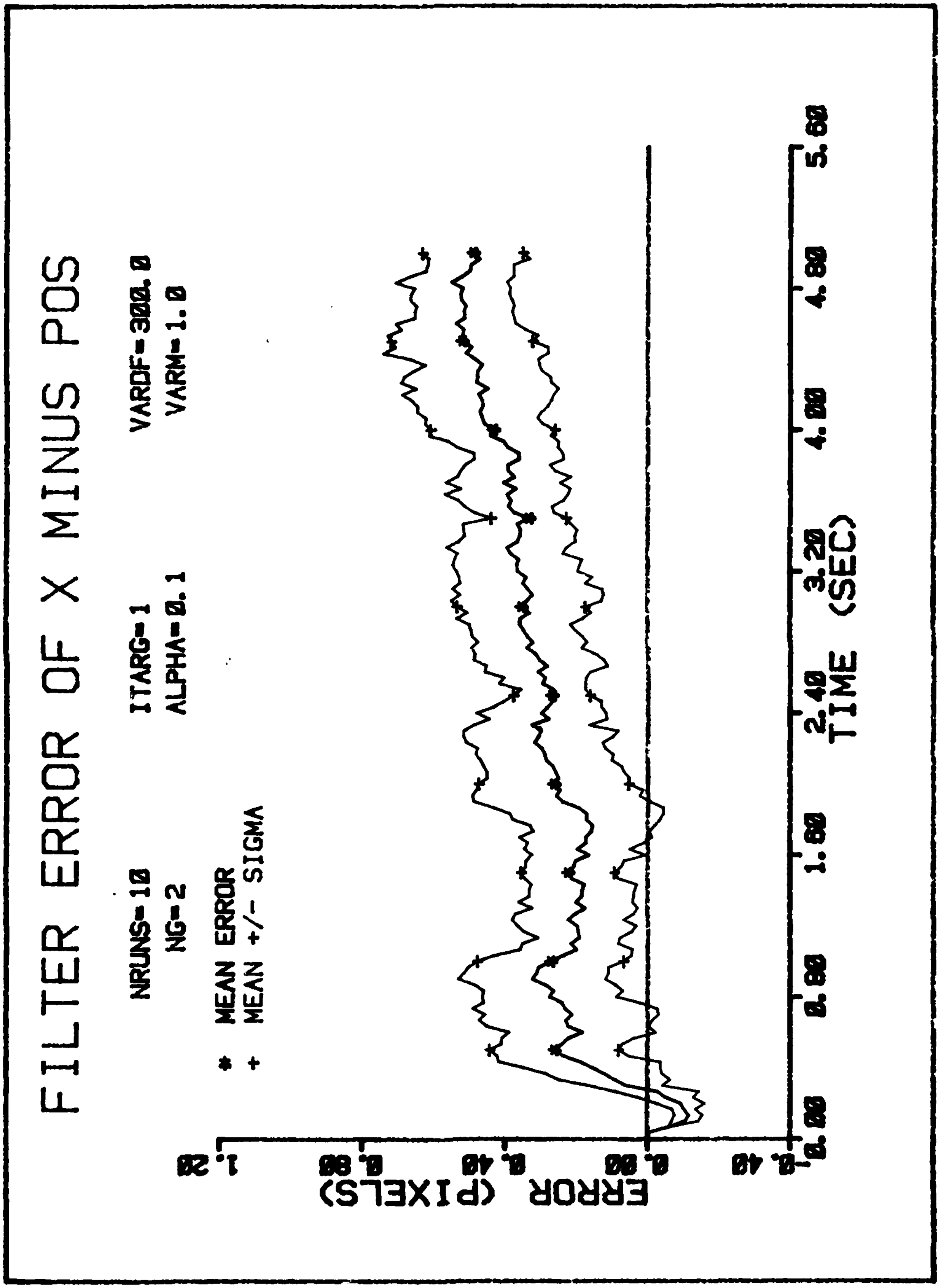


Figure E-120 Case 24 CTR Performance Plot

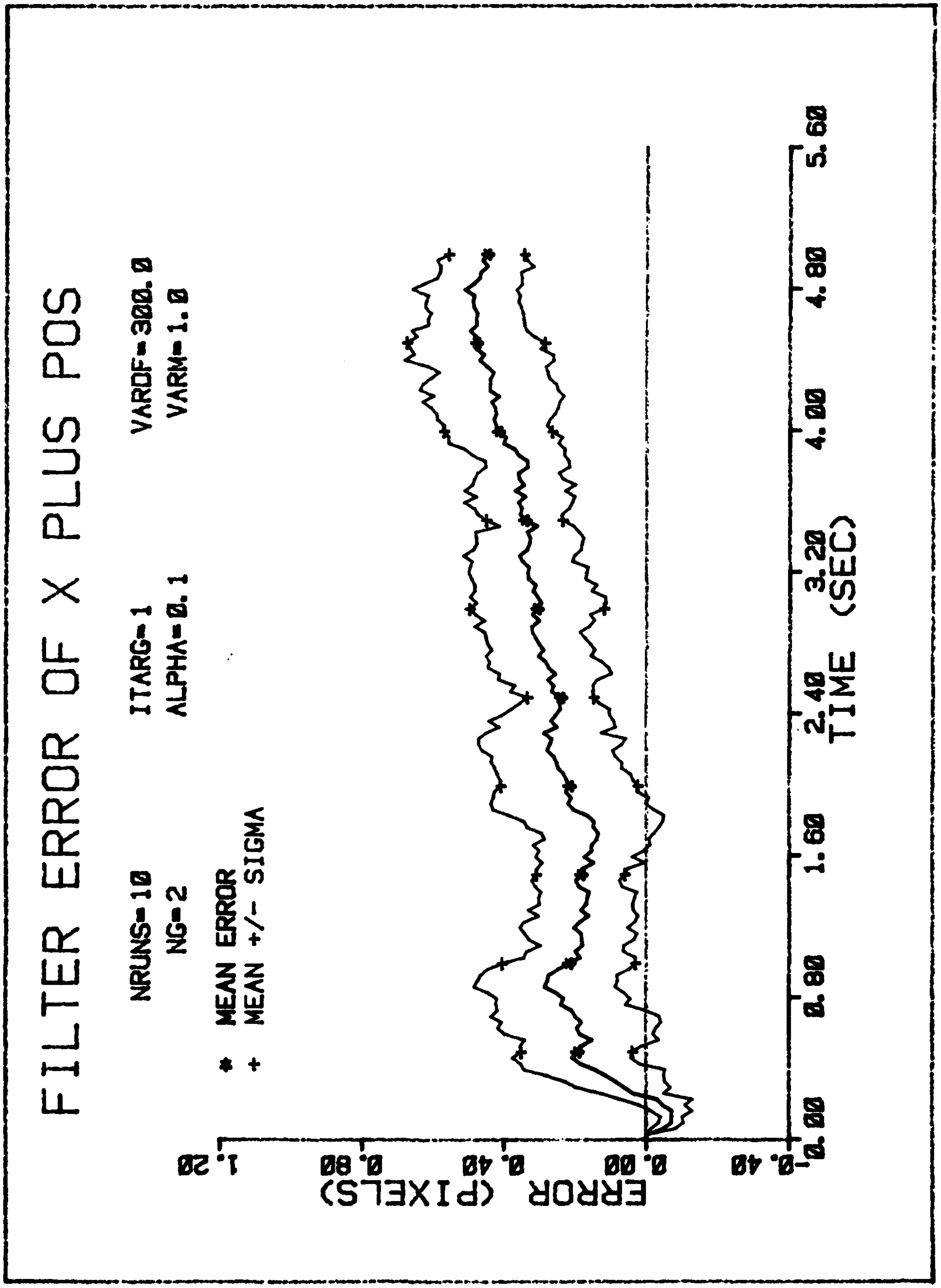


Figure E-121 Case 24 CTR Performance Plot

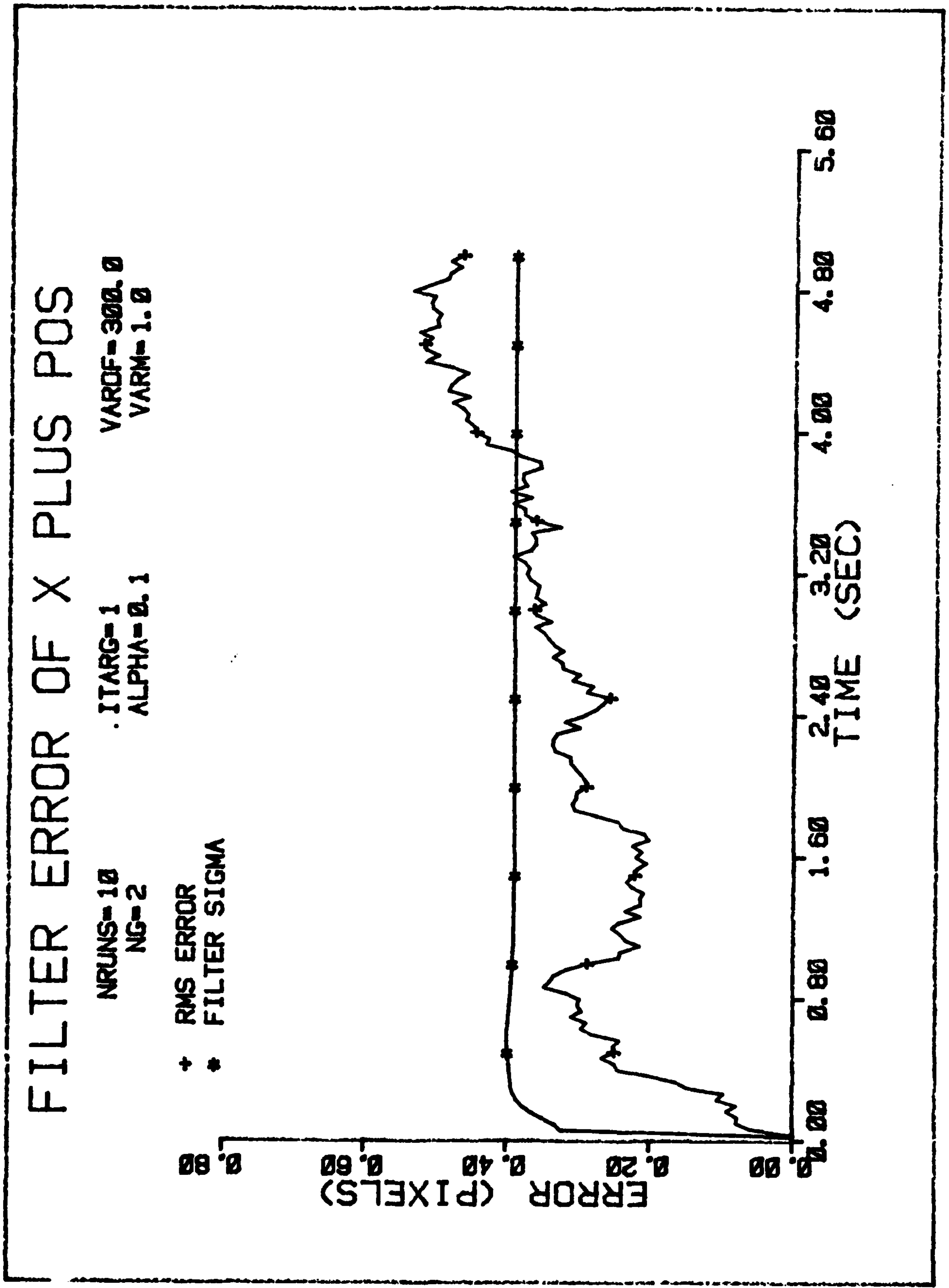


Figure E-122 Case 24 CTR Performance Plot

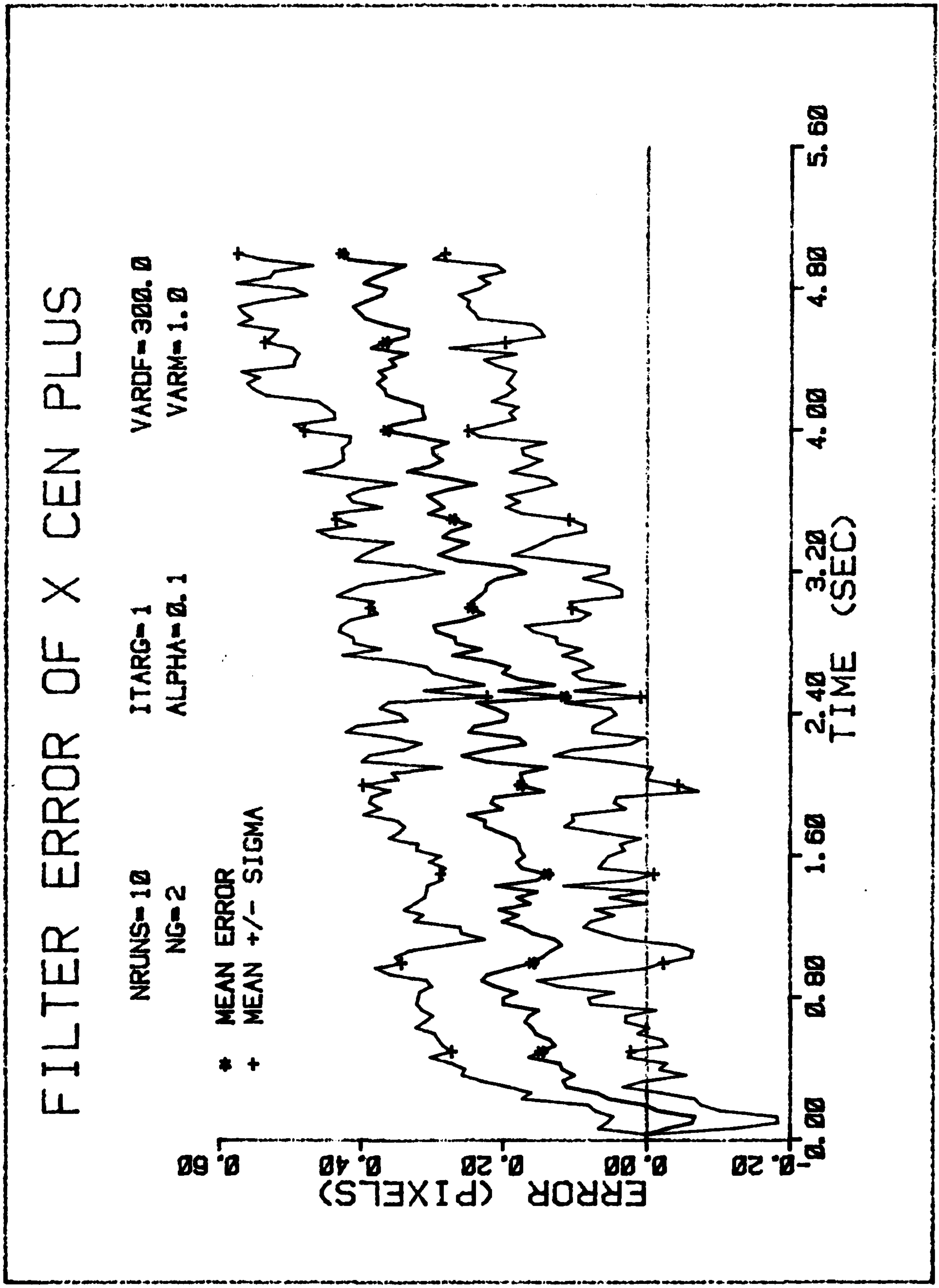


Figure E-123 Case 24 CTR Performance Plot

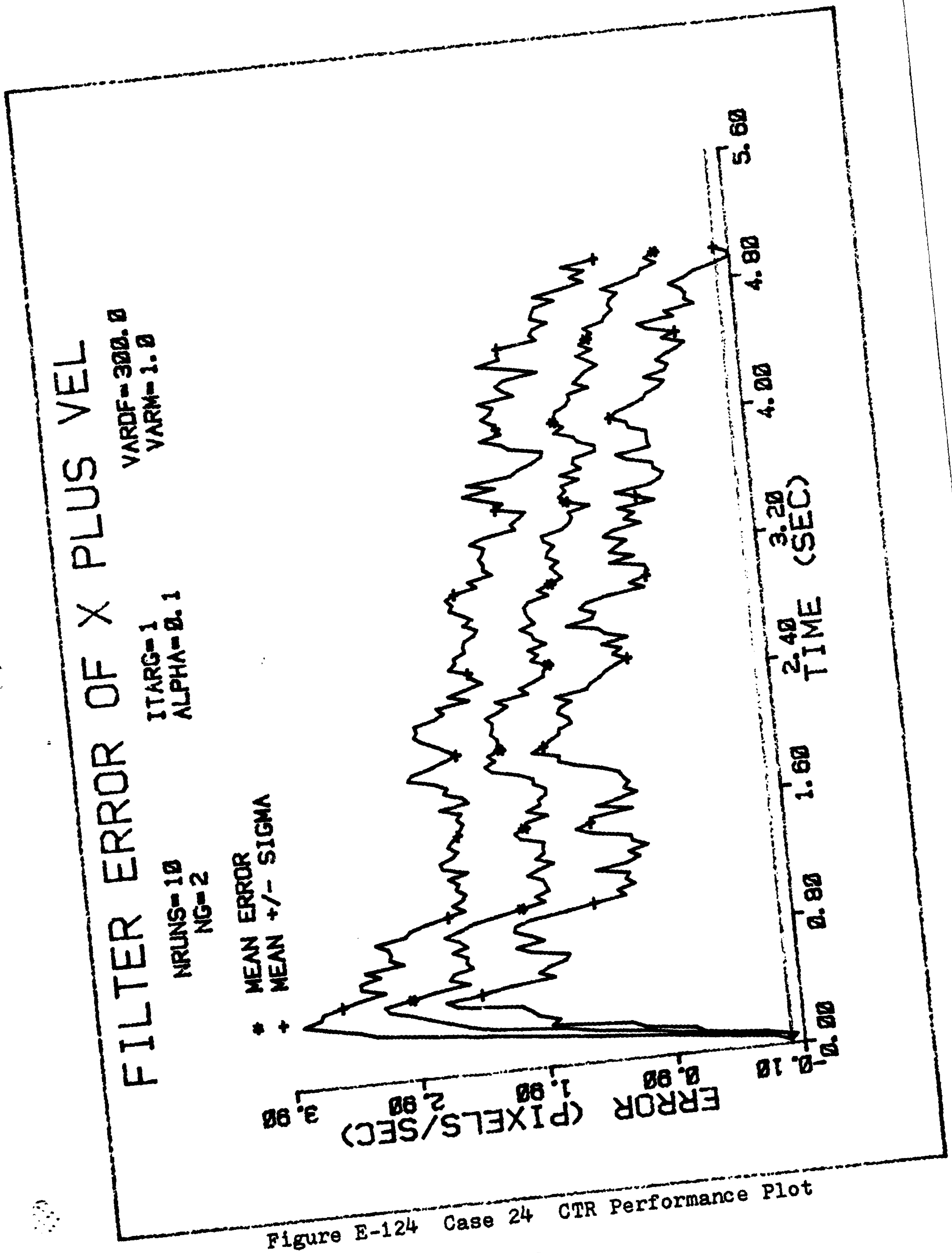


Figure E-124 Case 24 CTR Performance Plot

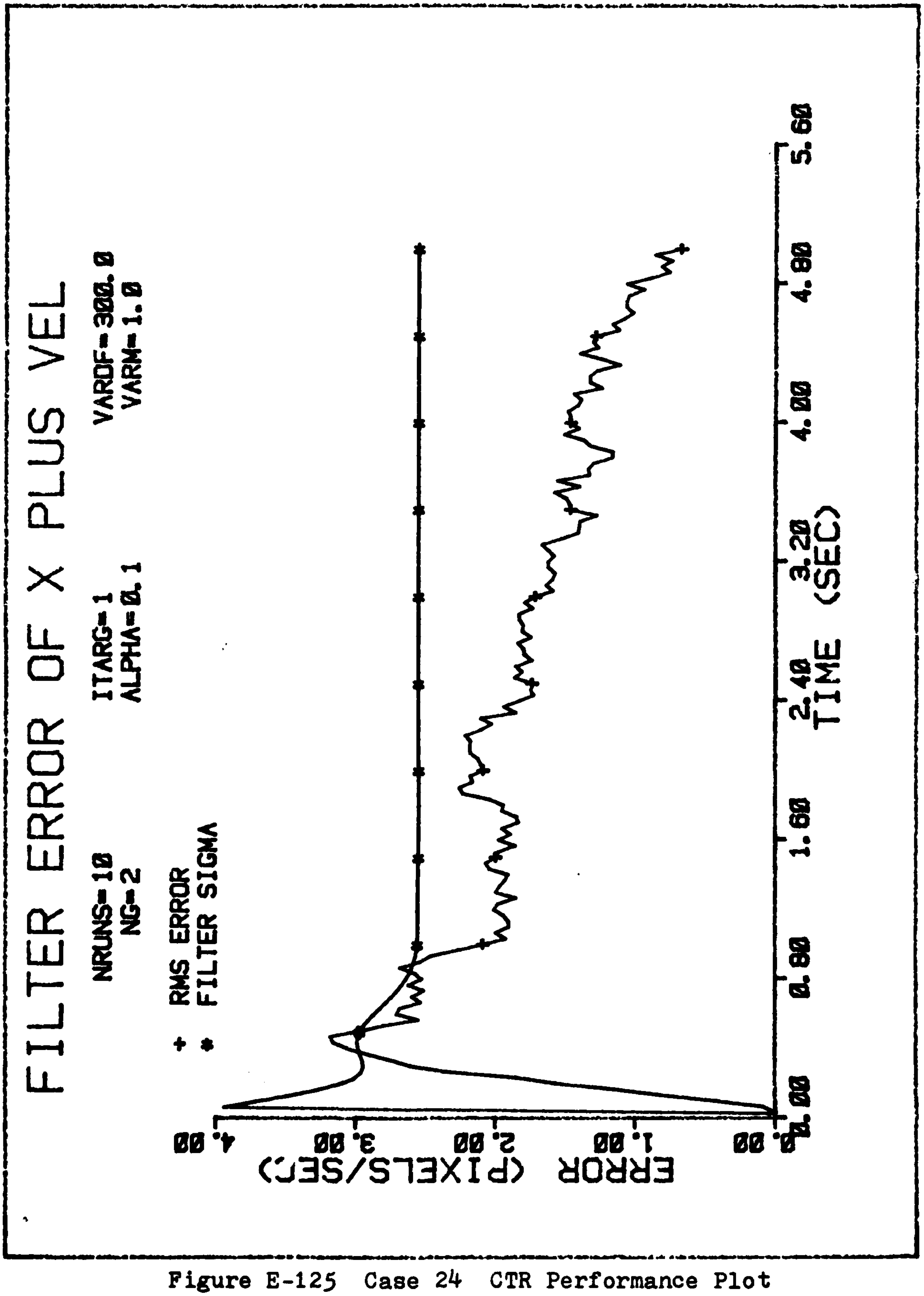


Figure E-125 Case 24 CTR Performance Plot

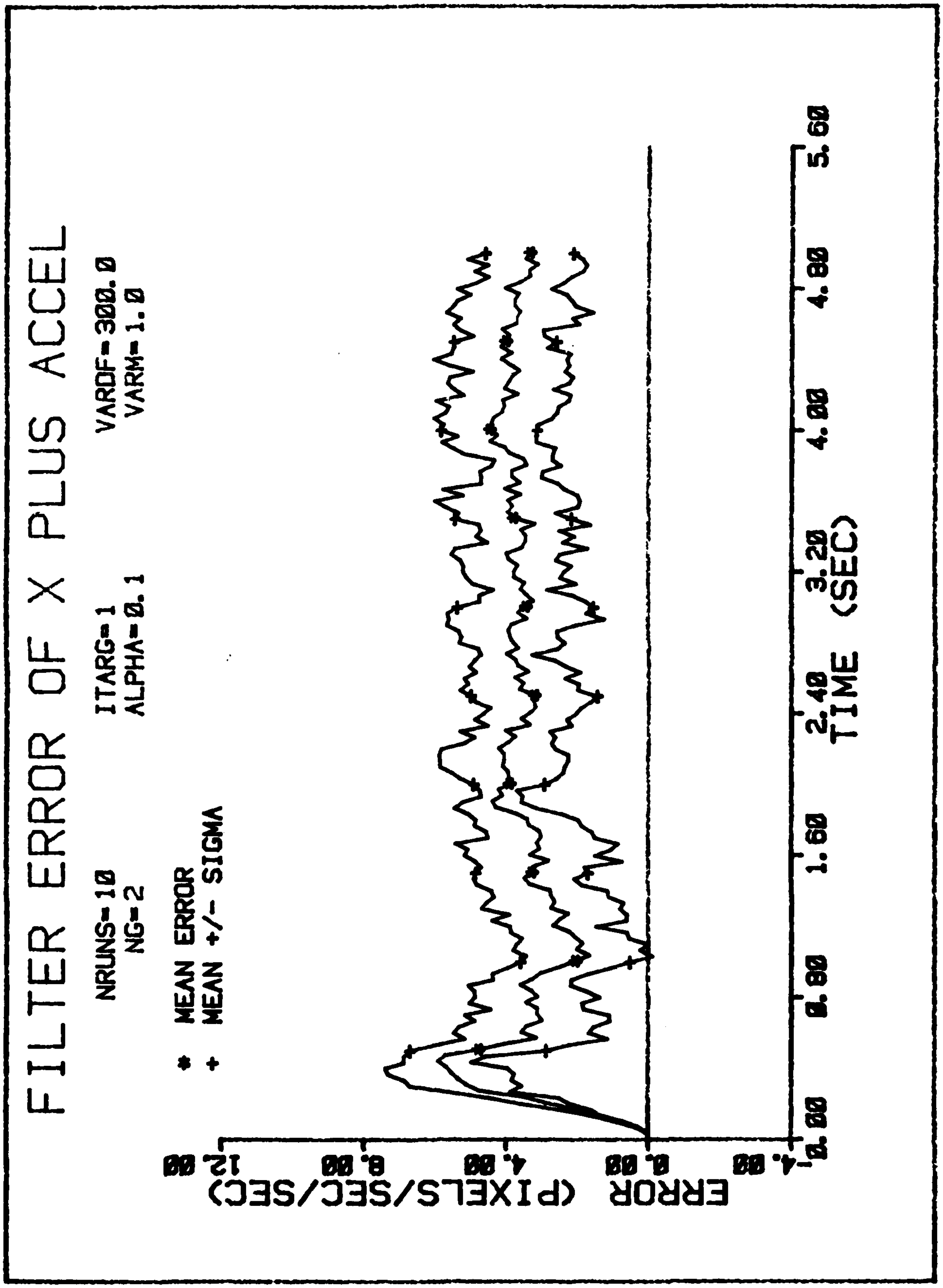


Figure E-126 Case 24 CTR Performance Plot

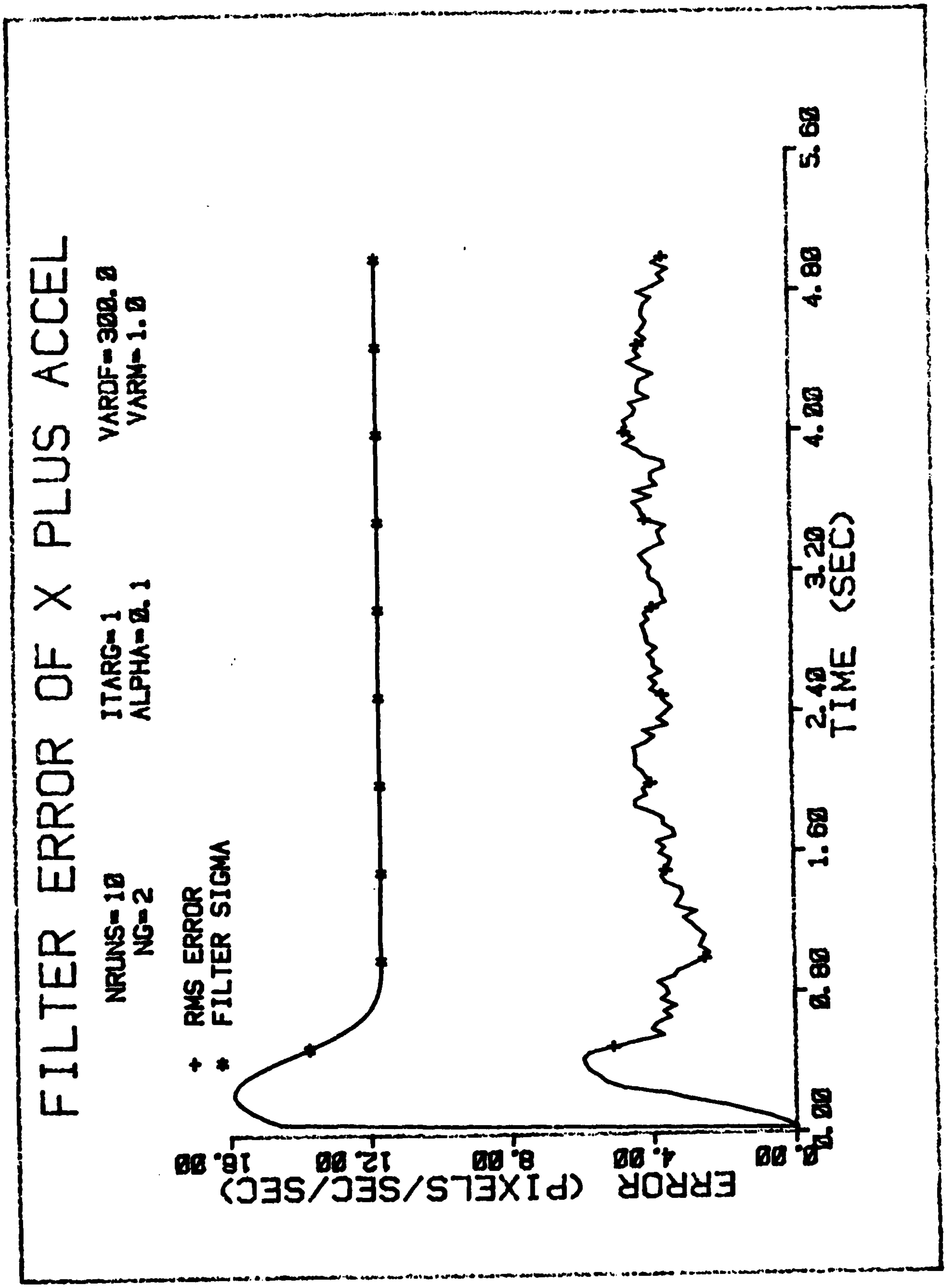


Figure E-127 Case 24 CTR Performance Plot

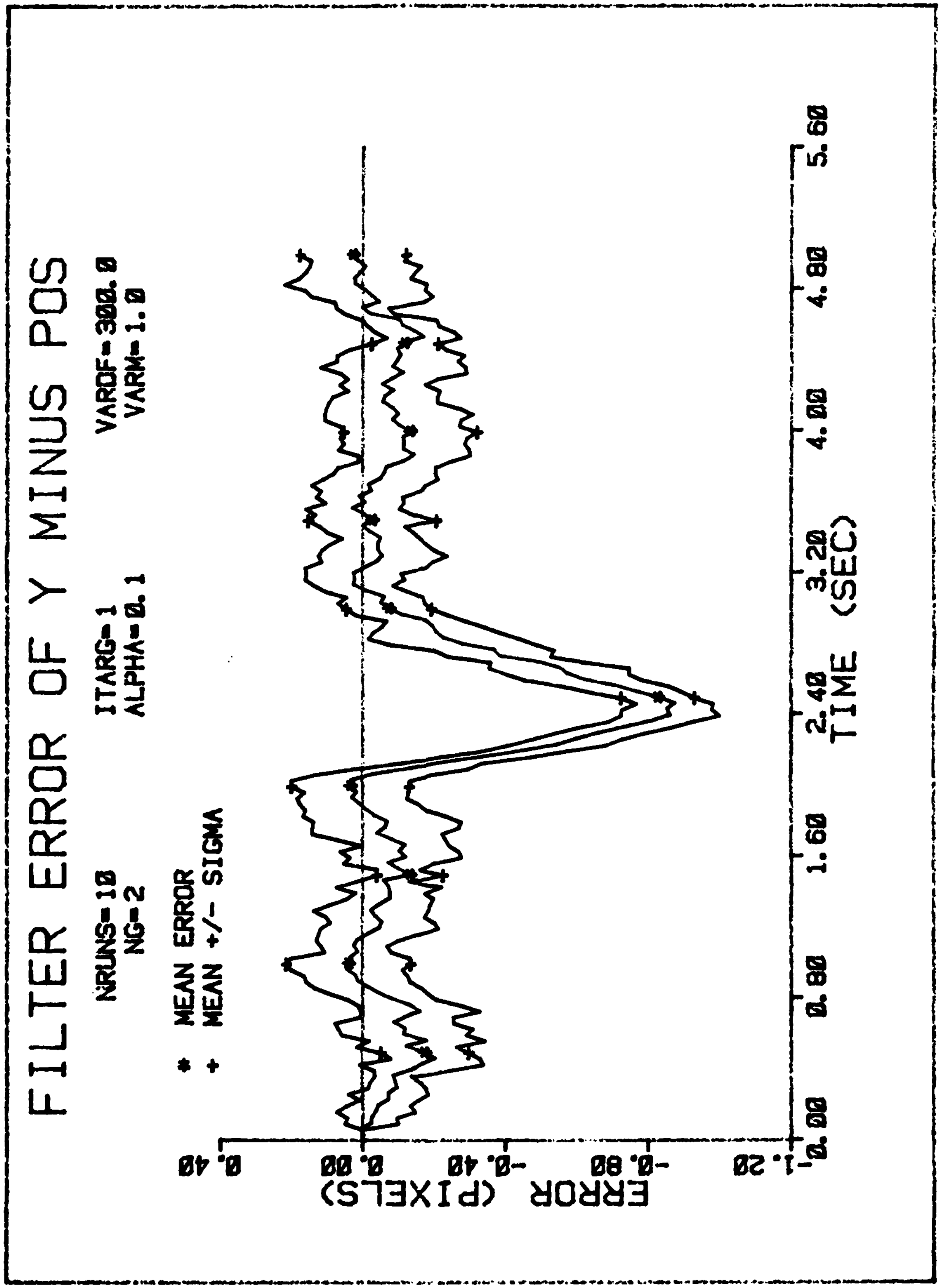


Figure E-128 Case 24 CTR Performance Plot

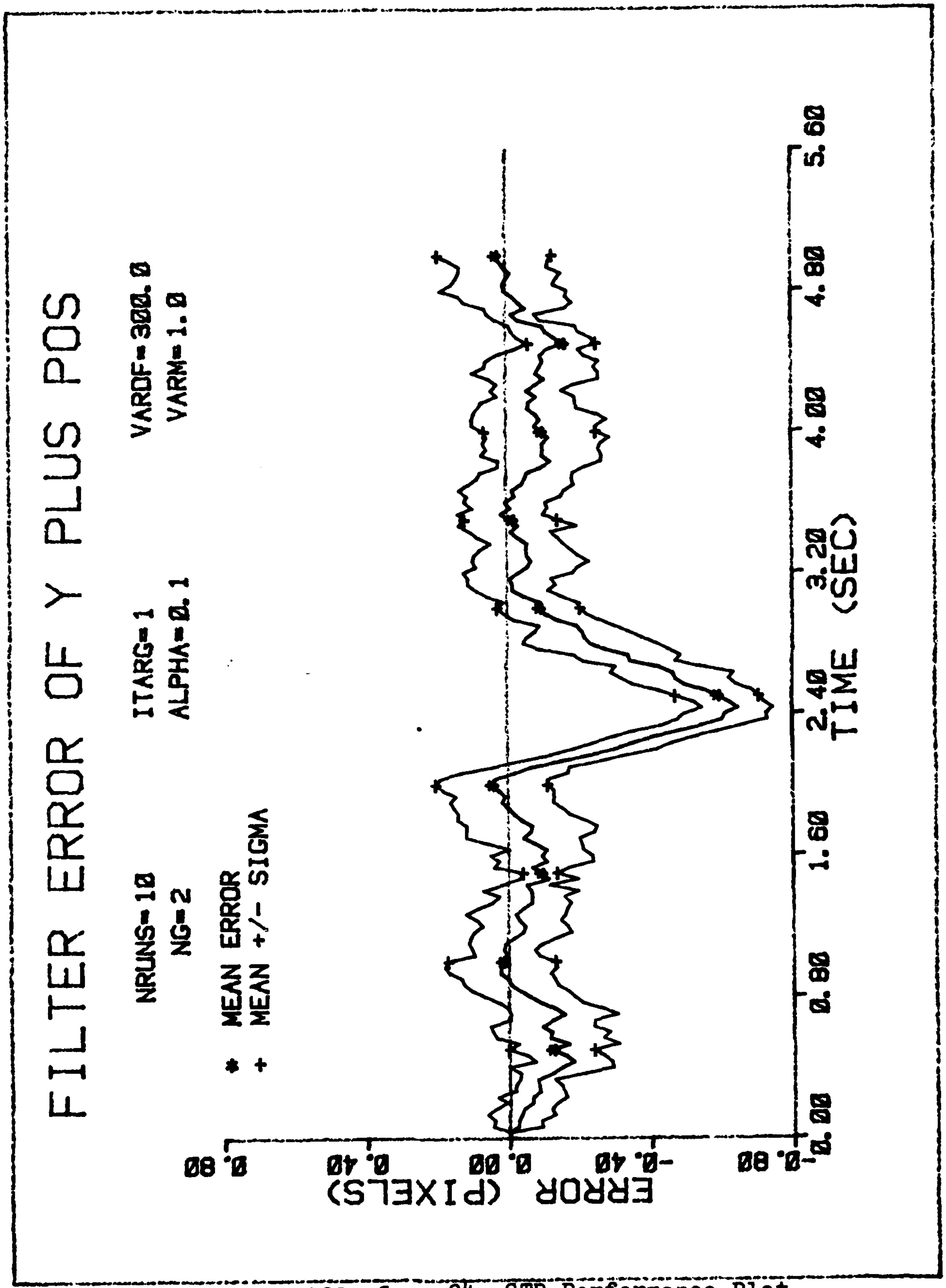


Figure E-129 Case 24 CTR Performance Plot

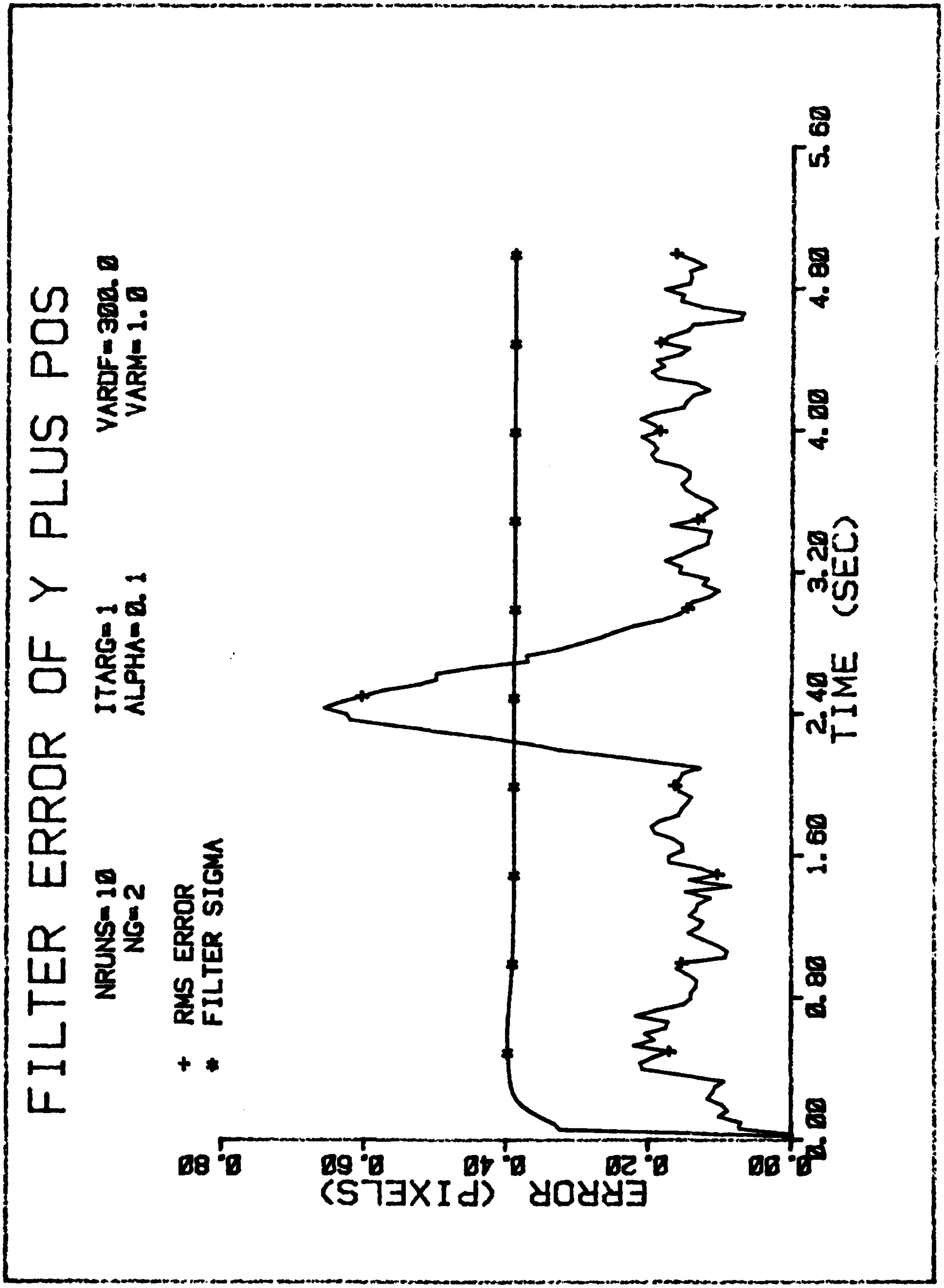


Figure E-130 Case 24 CTR Performance Plot

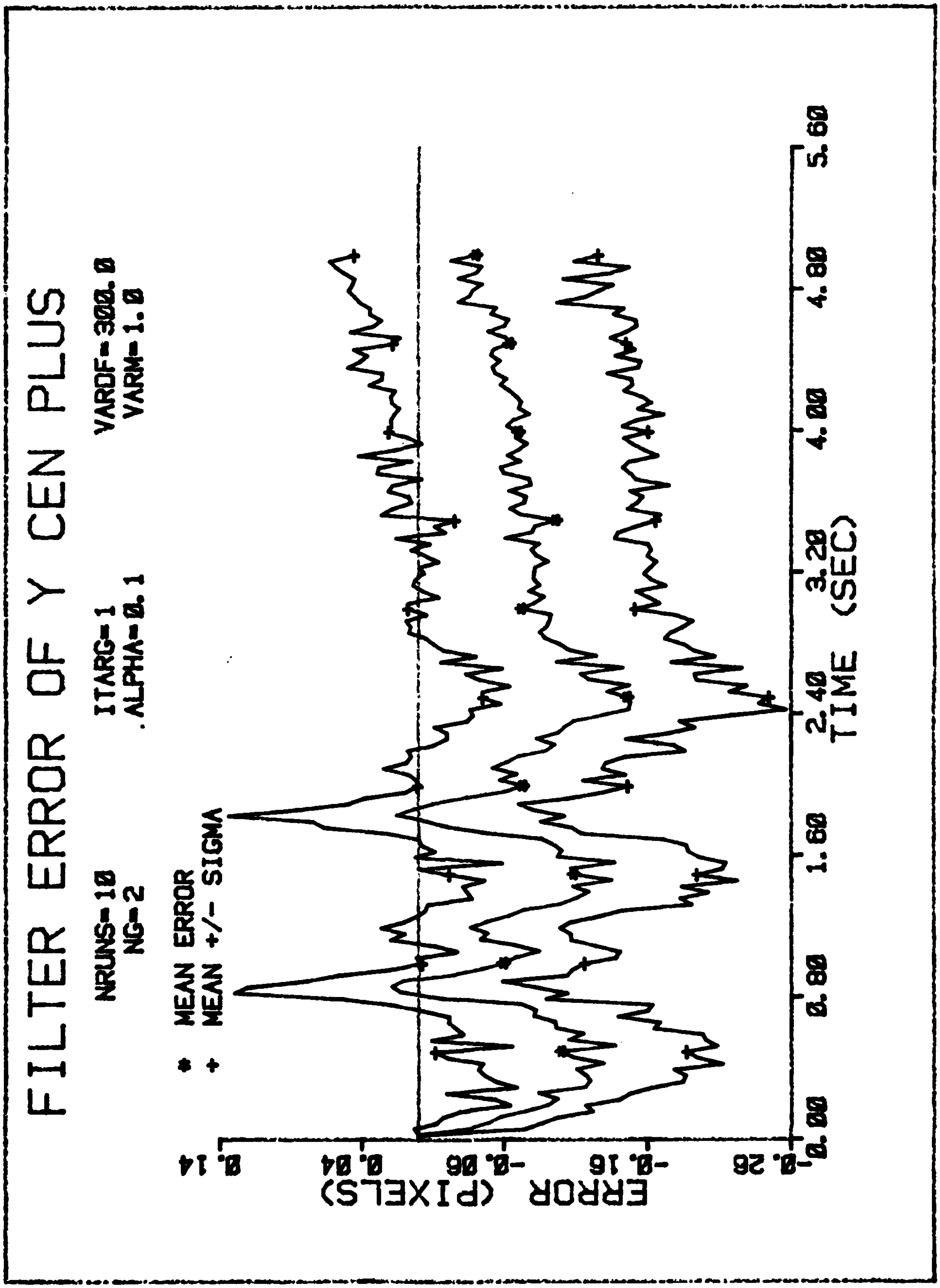


Figure E-131 Case 24 CTR Performance Plot

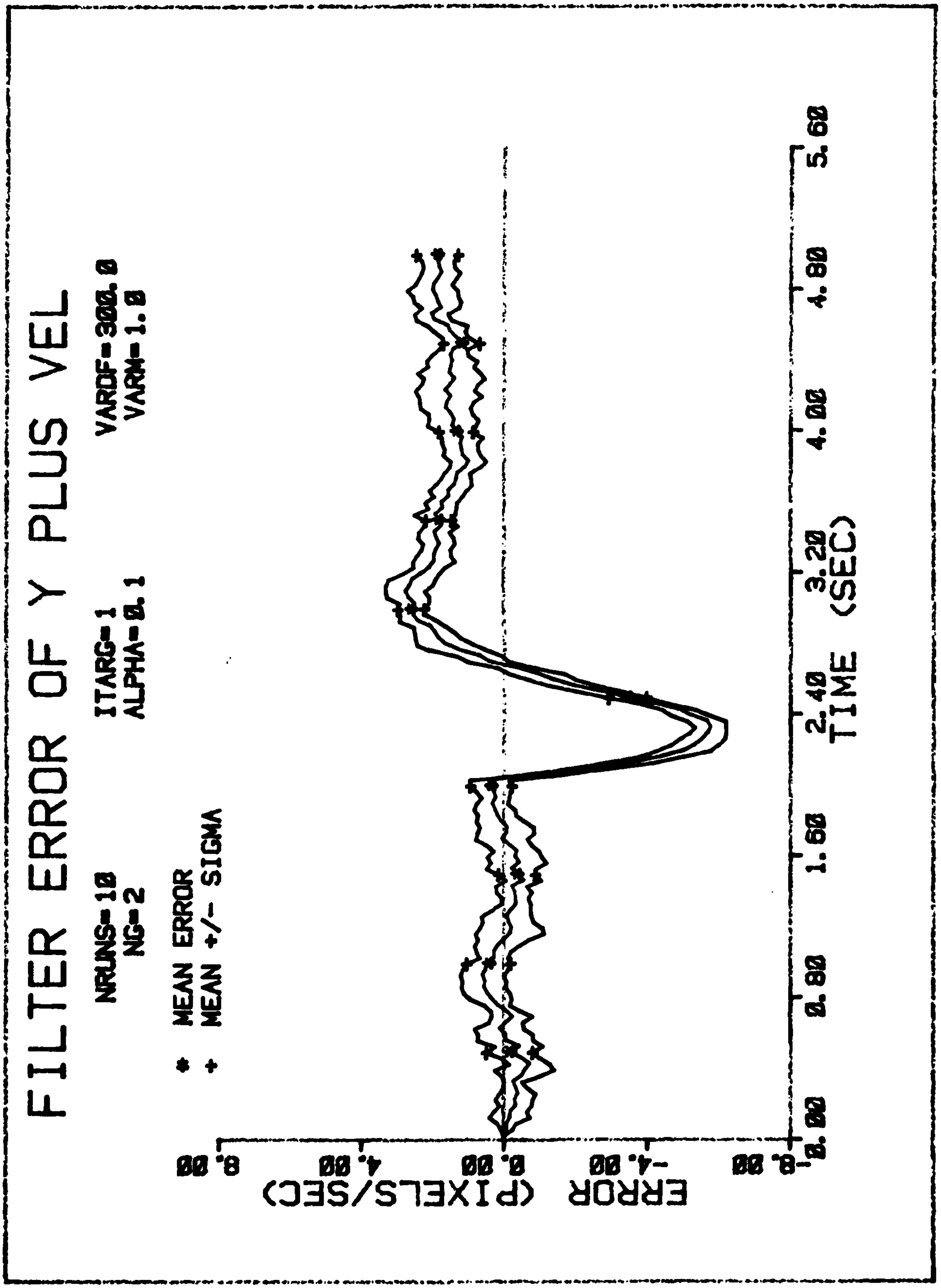


Figure E-132 Case 24 CTR Performance Plot

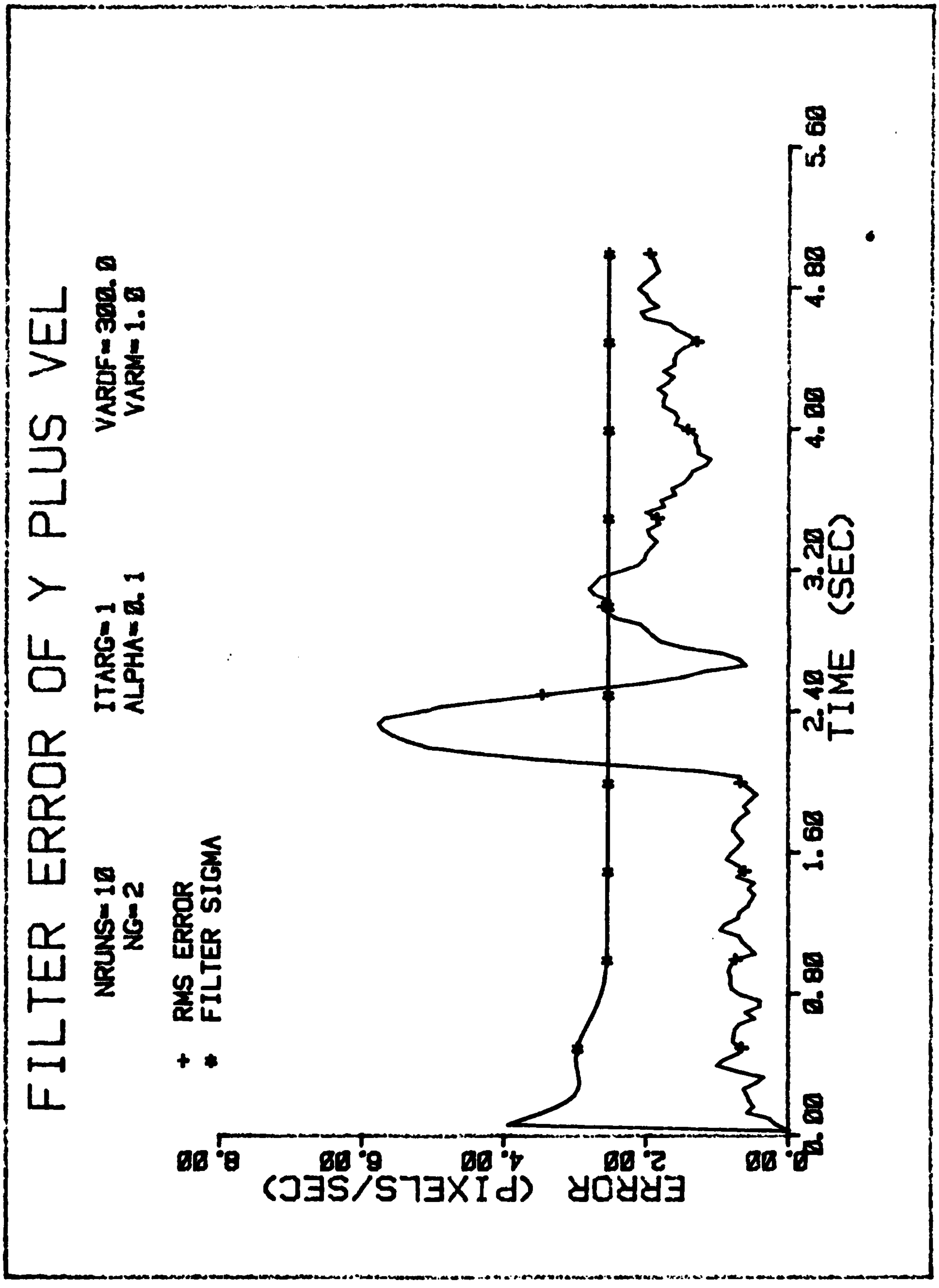


Figure E-133 Case 24 CTR Performance Plot

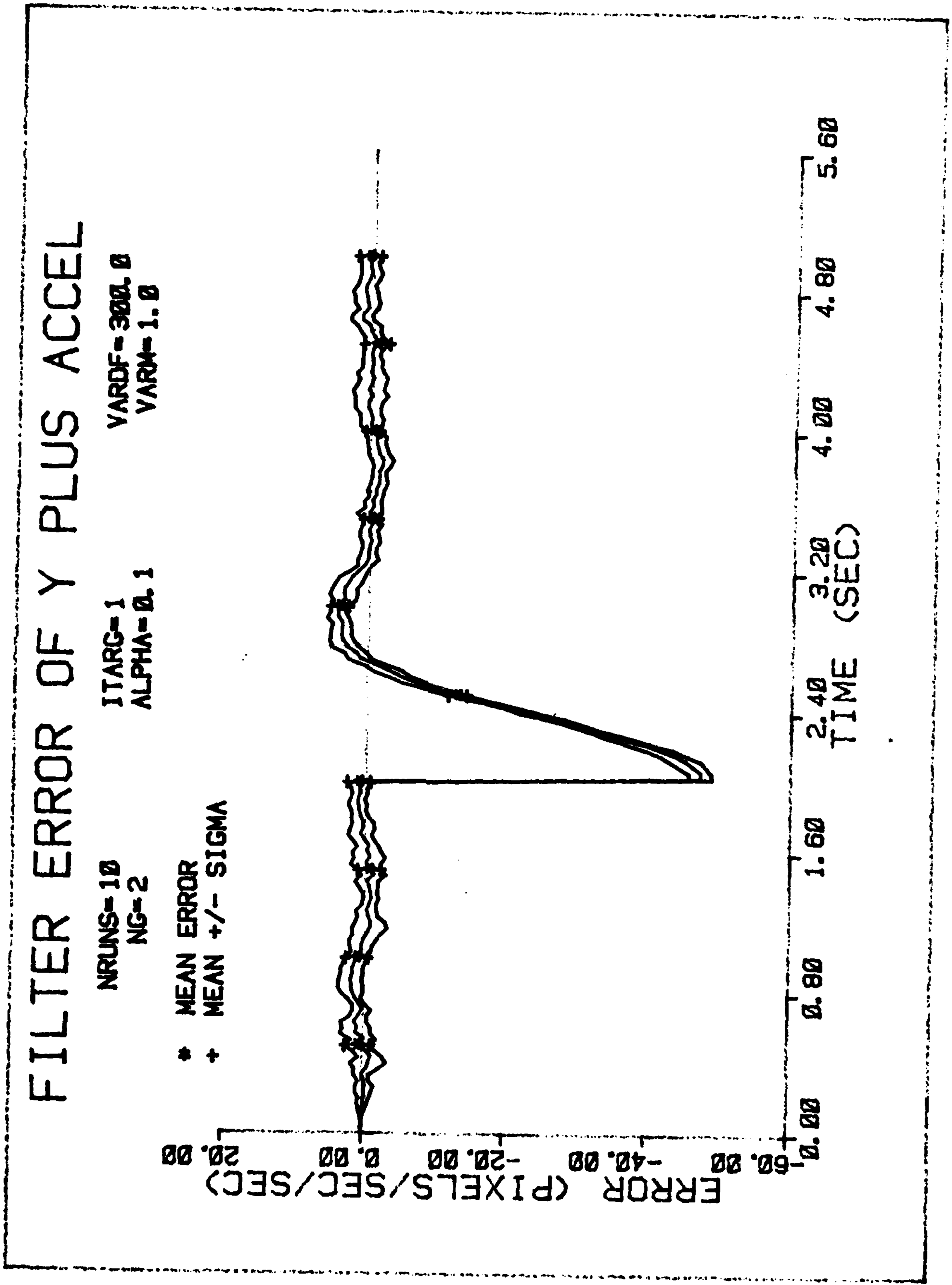


Figure E-134 Case 24 CTR Performance Plot

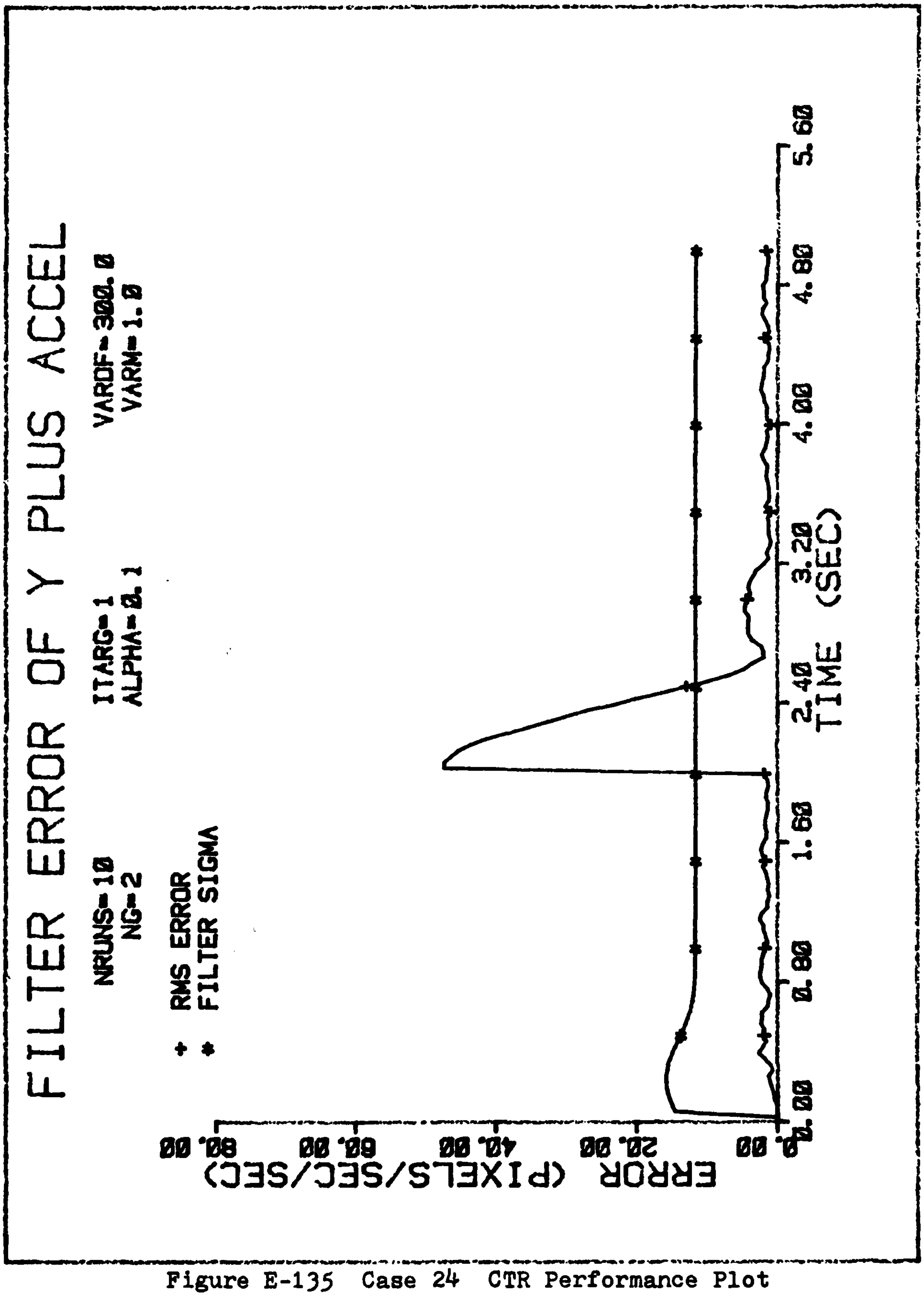
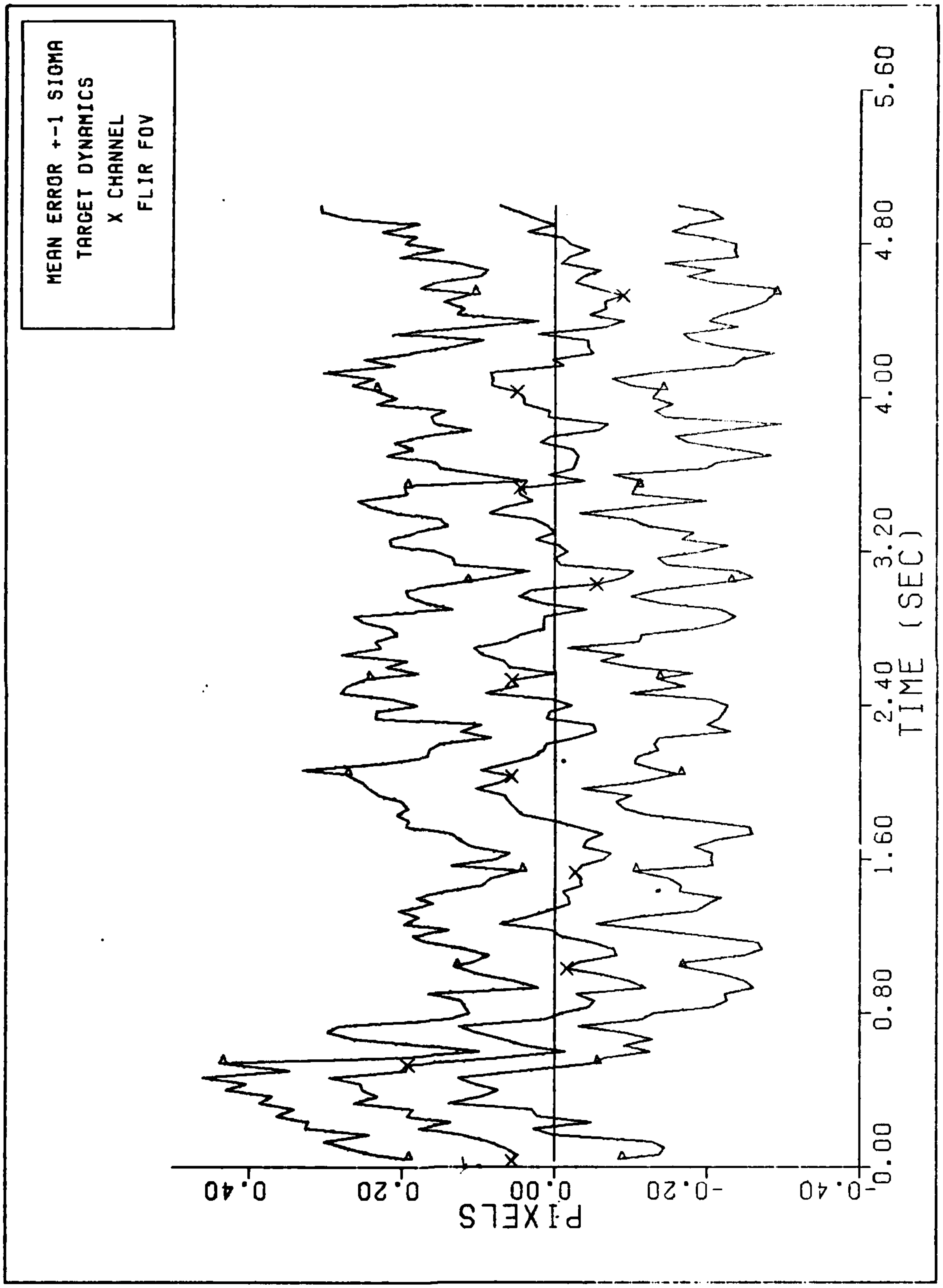


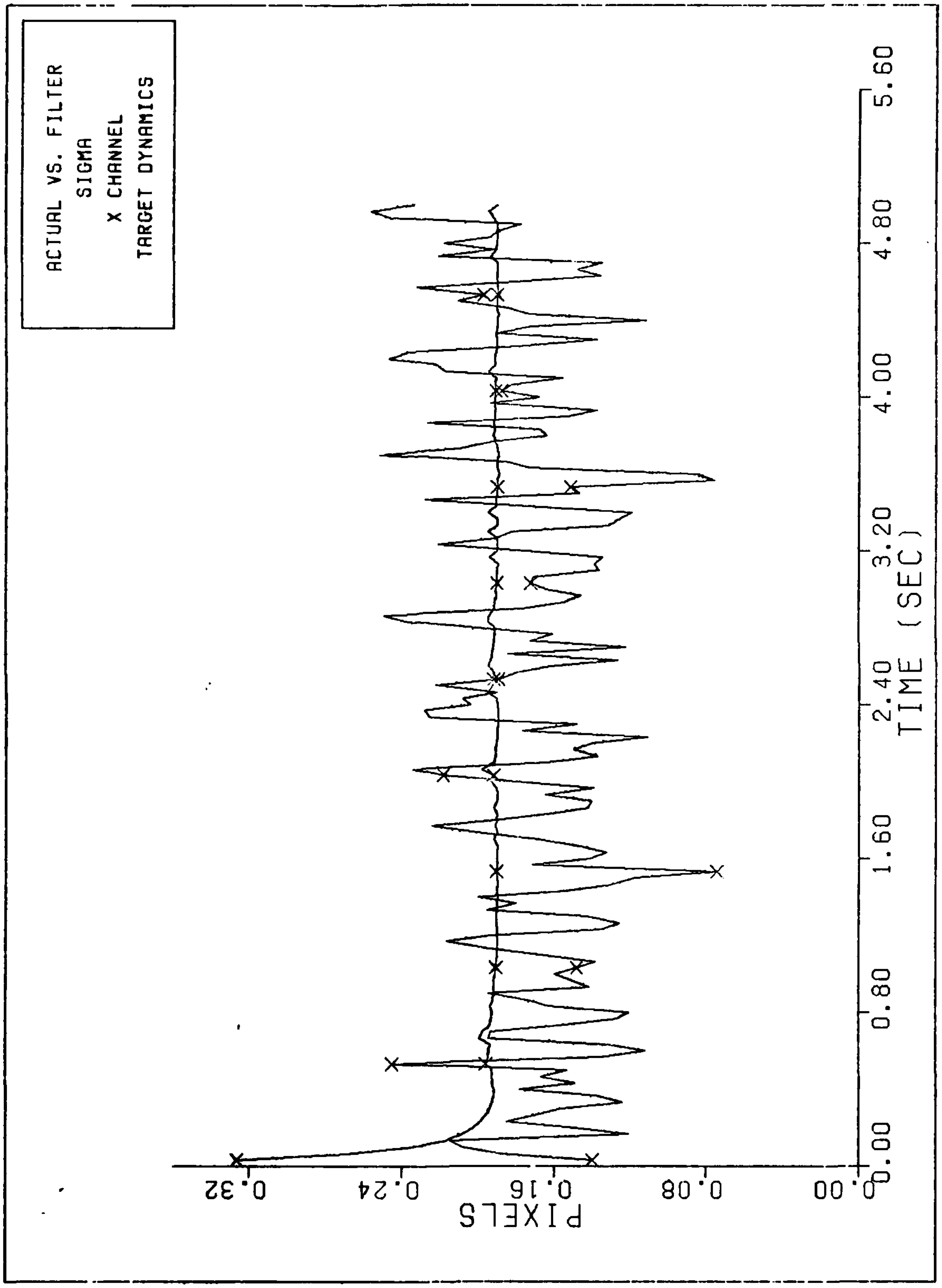
Figure E-135 Case 24 CTR Performance Plot

Appendix F

This appendix contains the plotted output for the previously developed Brownian Motion adaptive filter. Each plot is labled with the trajectory or maneuver that was performed during the simulation.

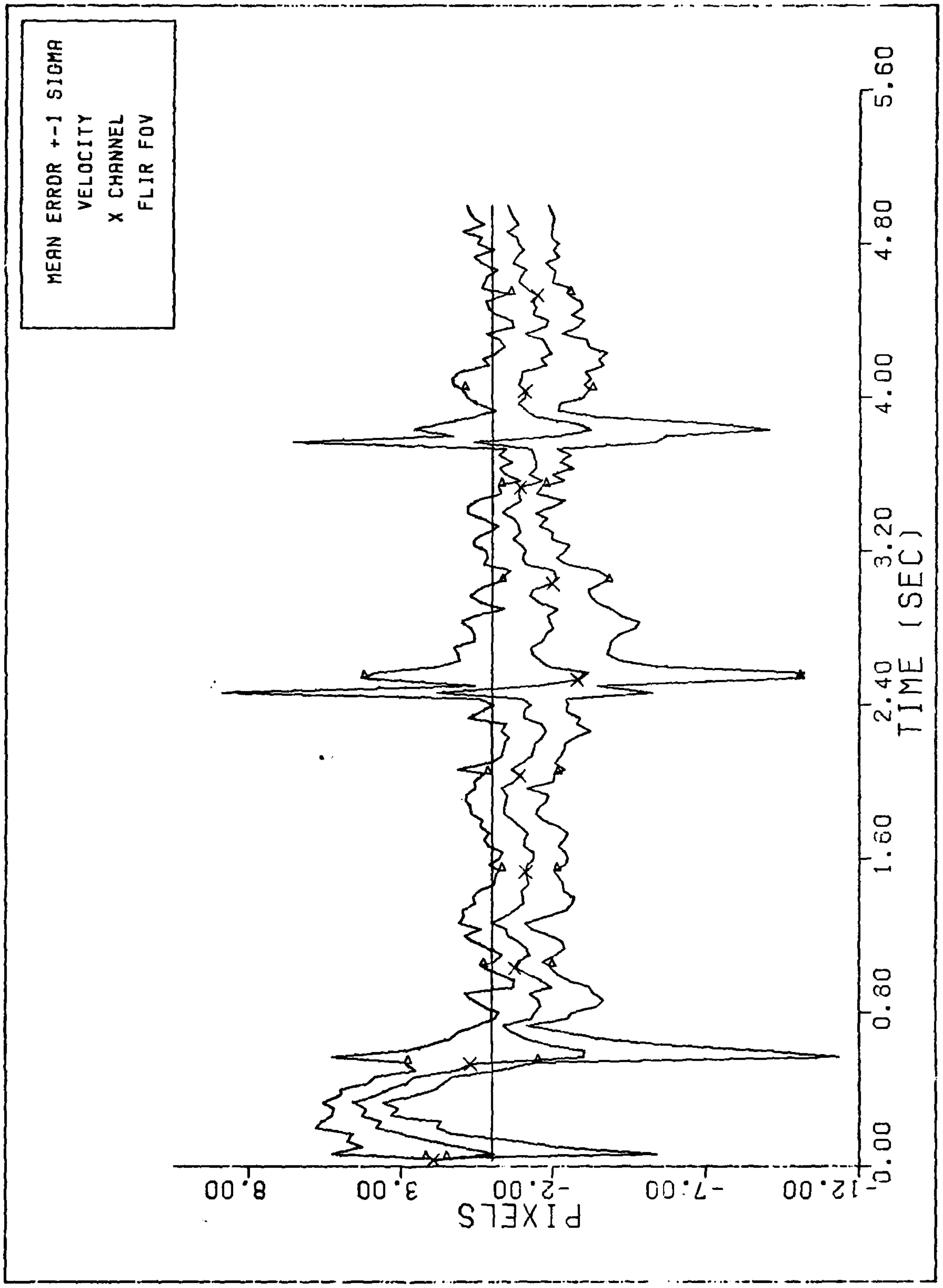


X CHANNEL DYNAMICS ERROR (S/N= 20)
Figure F-1 BM Performance Plot - Trajectory 3

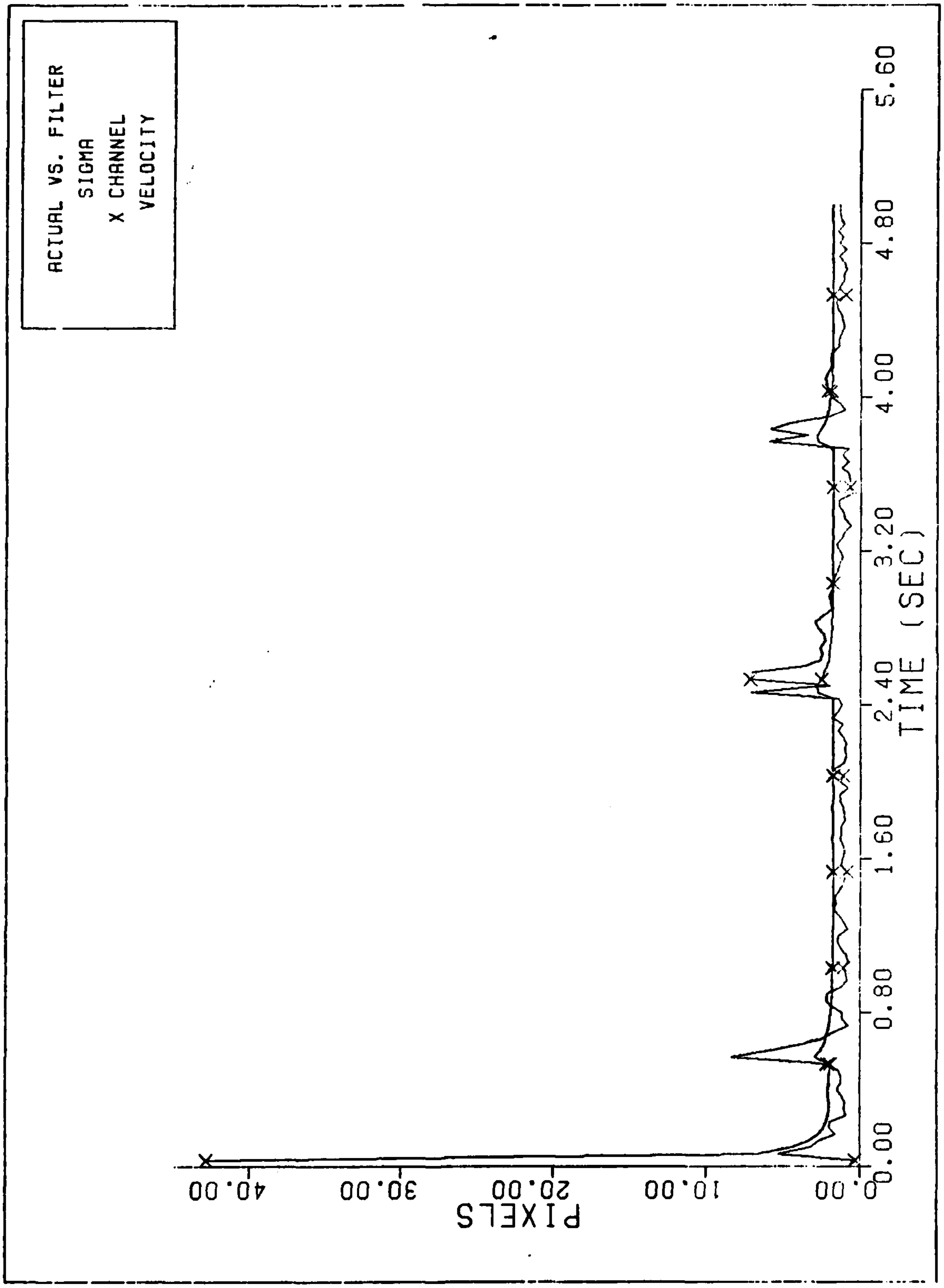


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

Figure F-2 BM Performance Plot - Trajectory 3

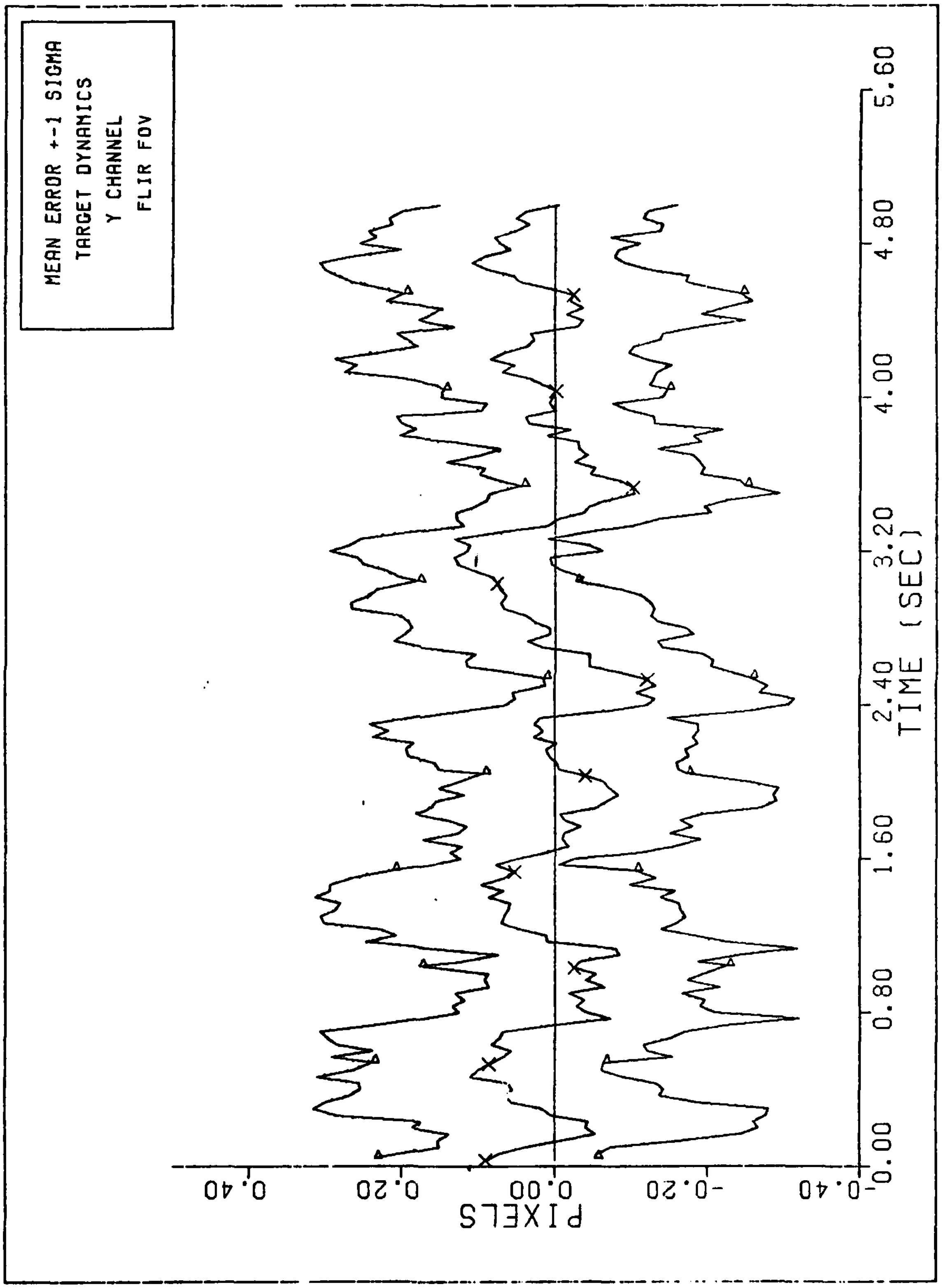


X CHANNEL VELOCITY ERROR (S/N=20)
Figure F-3 BM Performance Plot - Trajectory 3

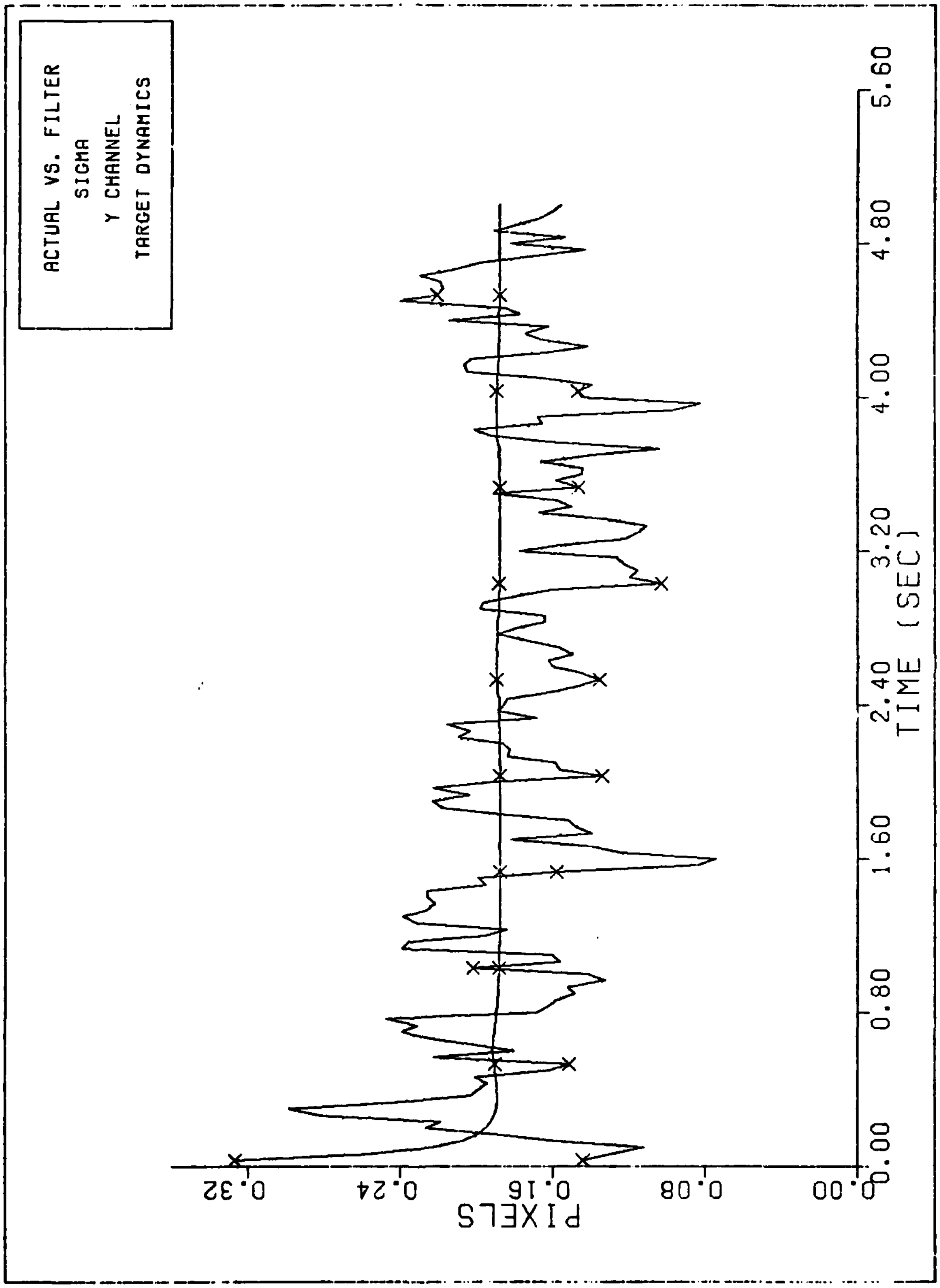


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

Figure F-4 BM Performance Plot - Trajectory 3

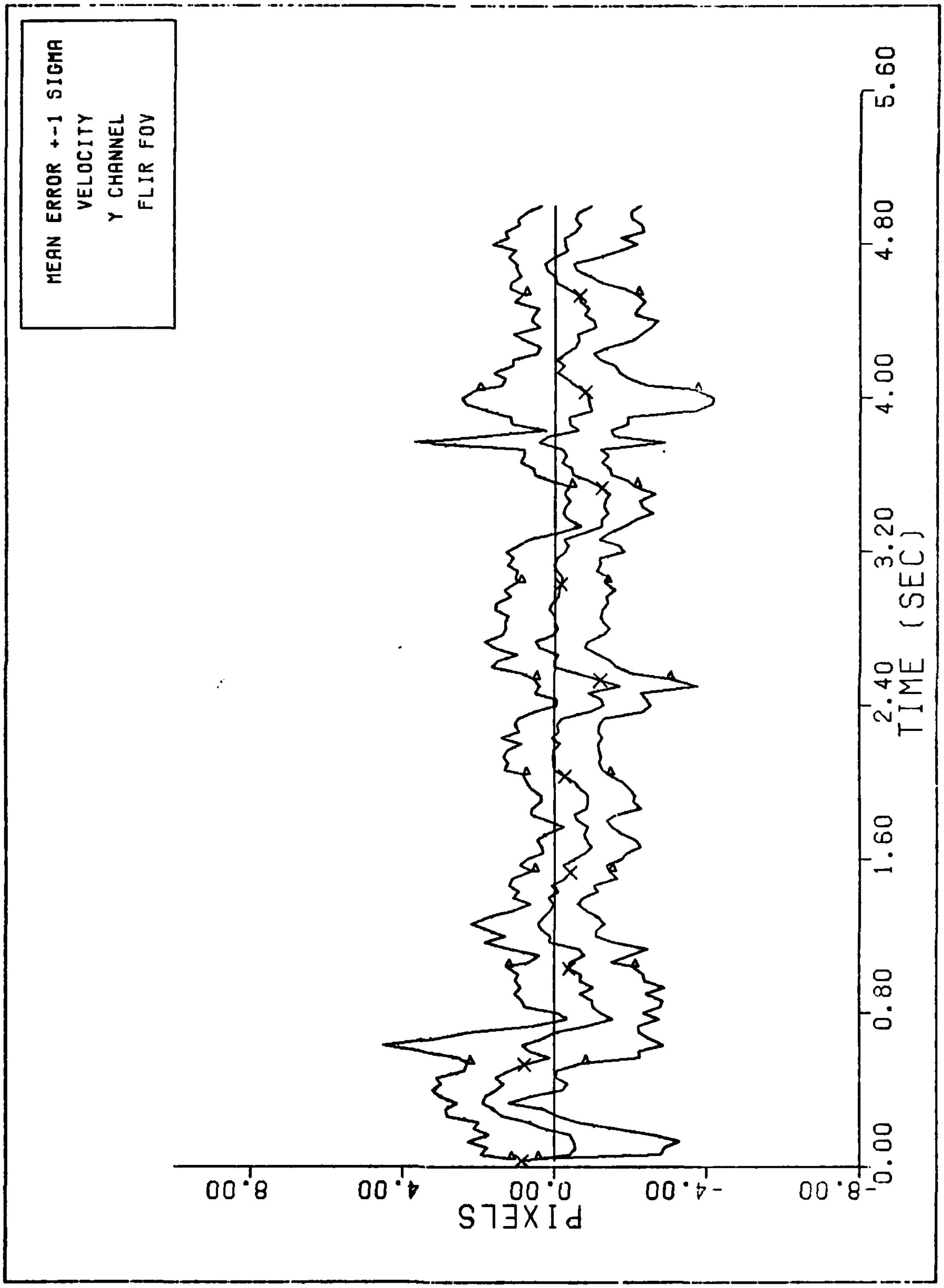


Y CHANNEL DYNAMICS ERROR (S/N=20)
Figure F-5 BM Performance Plot - Trajectory 3



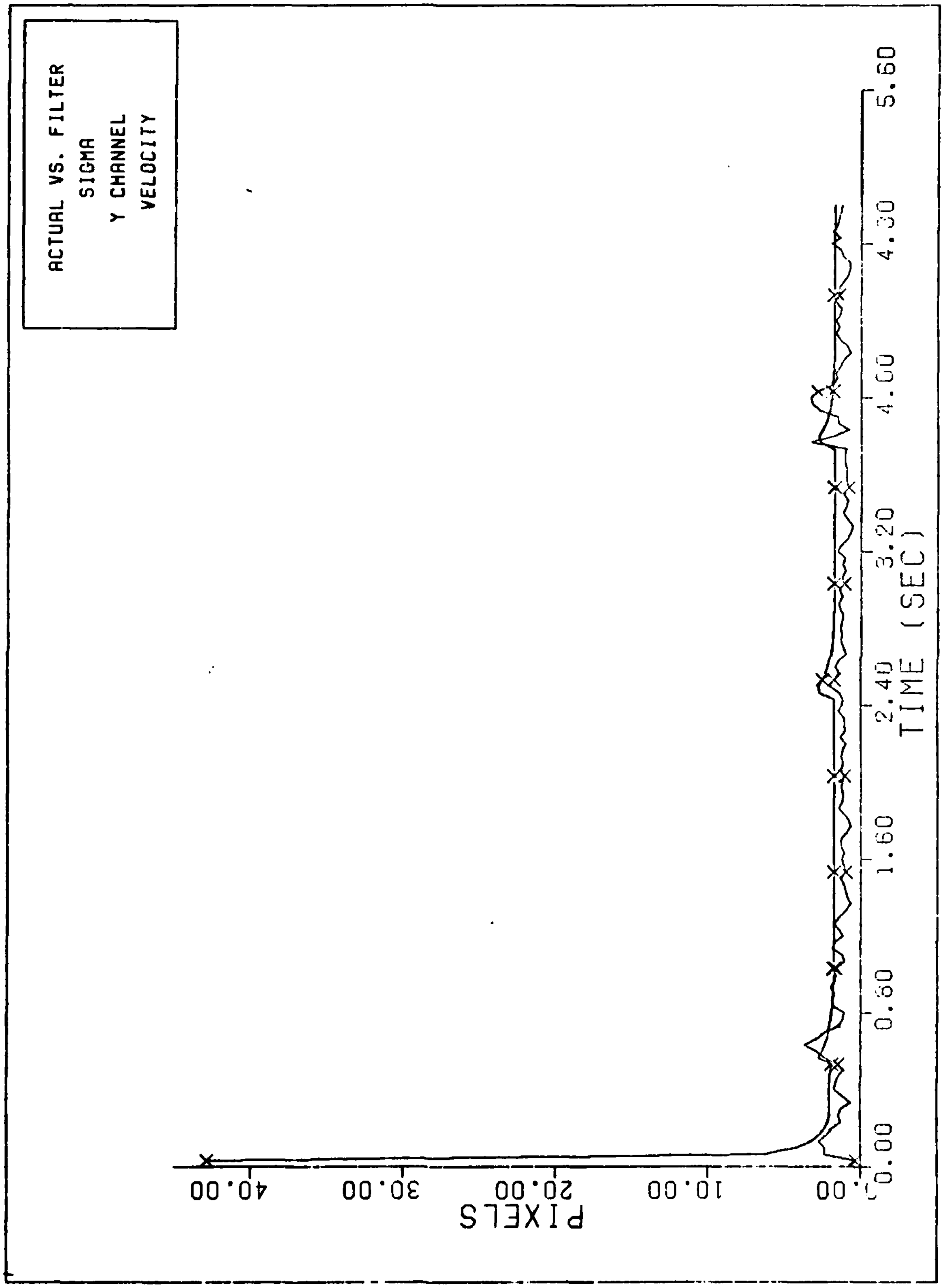
FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

Figure F-6 BM Performance Plot - Trajectory 3

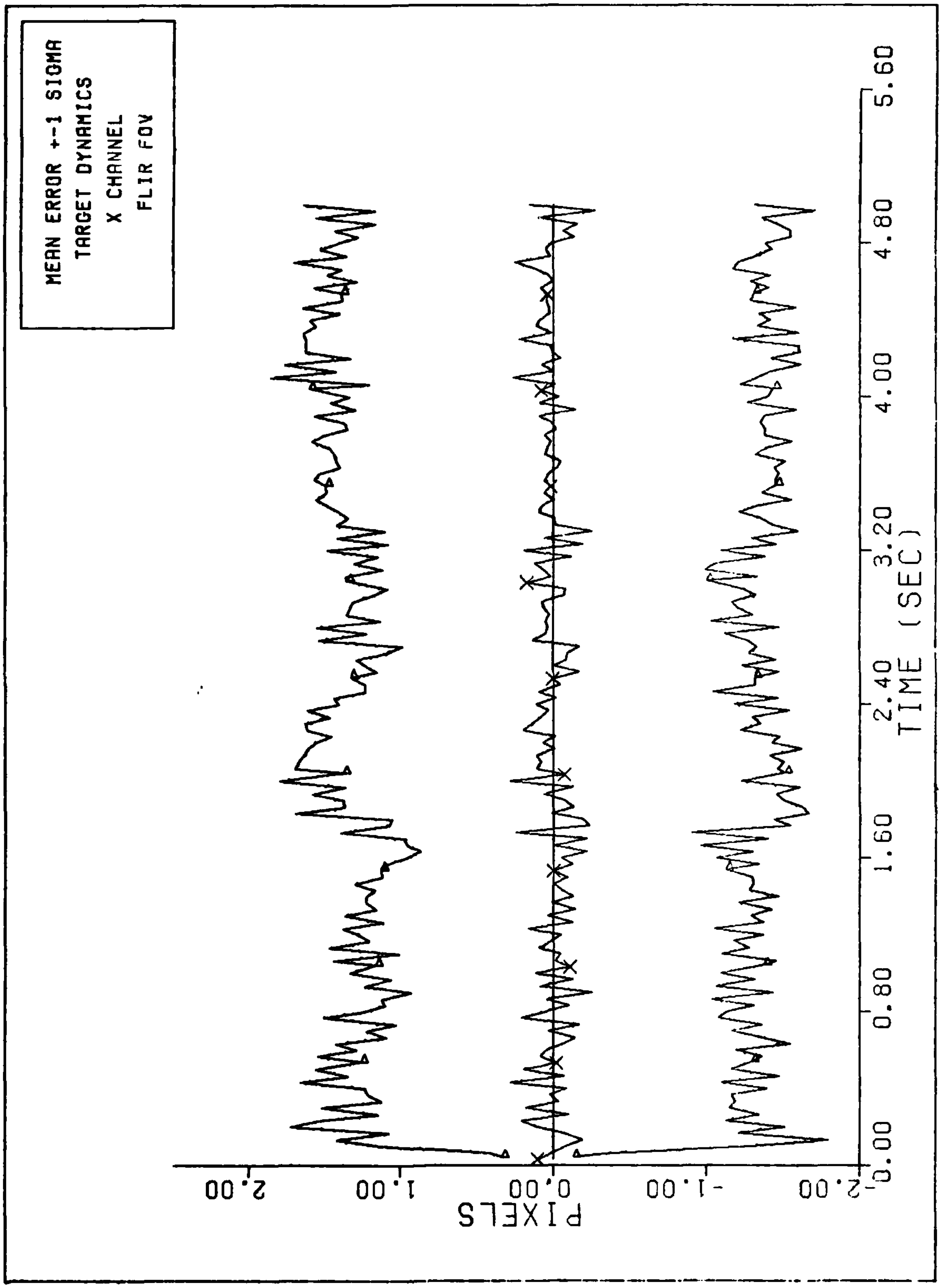


Y CHANNEL VELOCITY ERROR (S/N= 20)

Figure F-7 BM Performance Plot - Trajectory 3

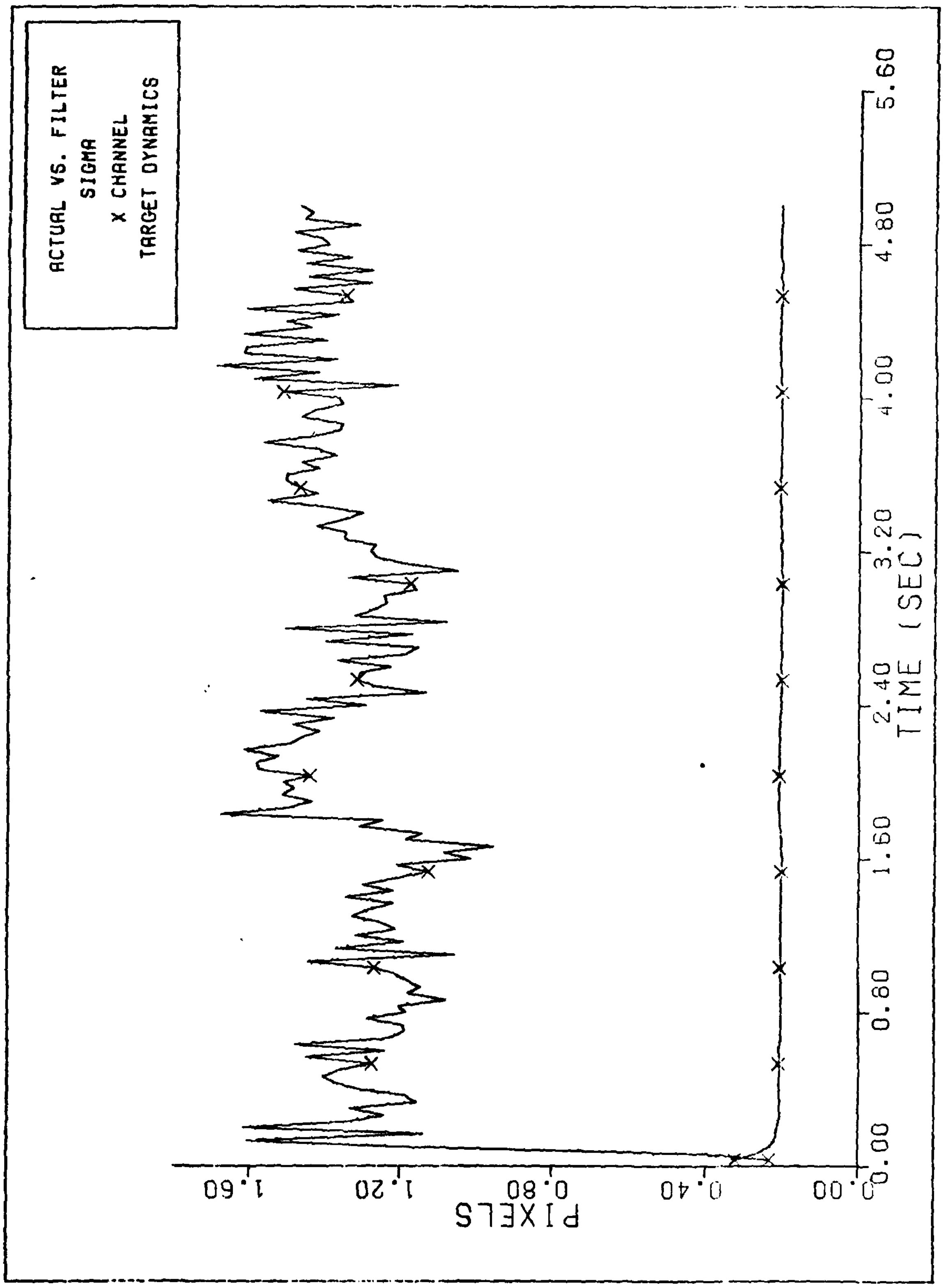


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)
Figure F-8 BM Performance Plot - Trajectory 3



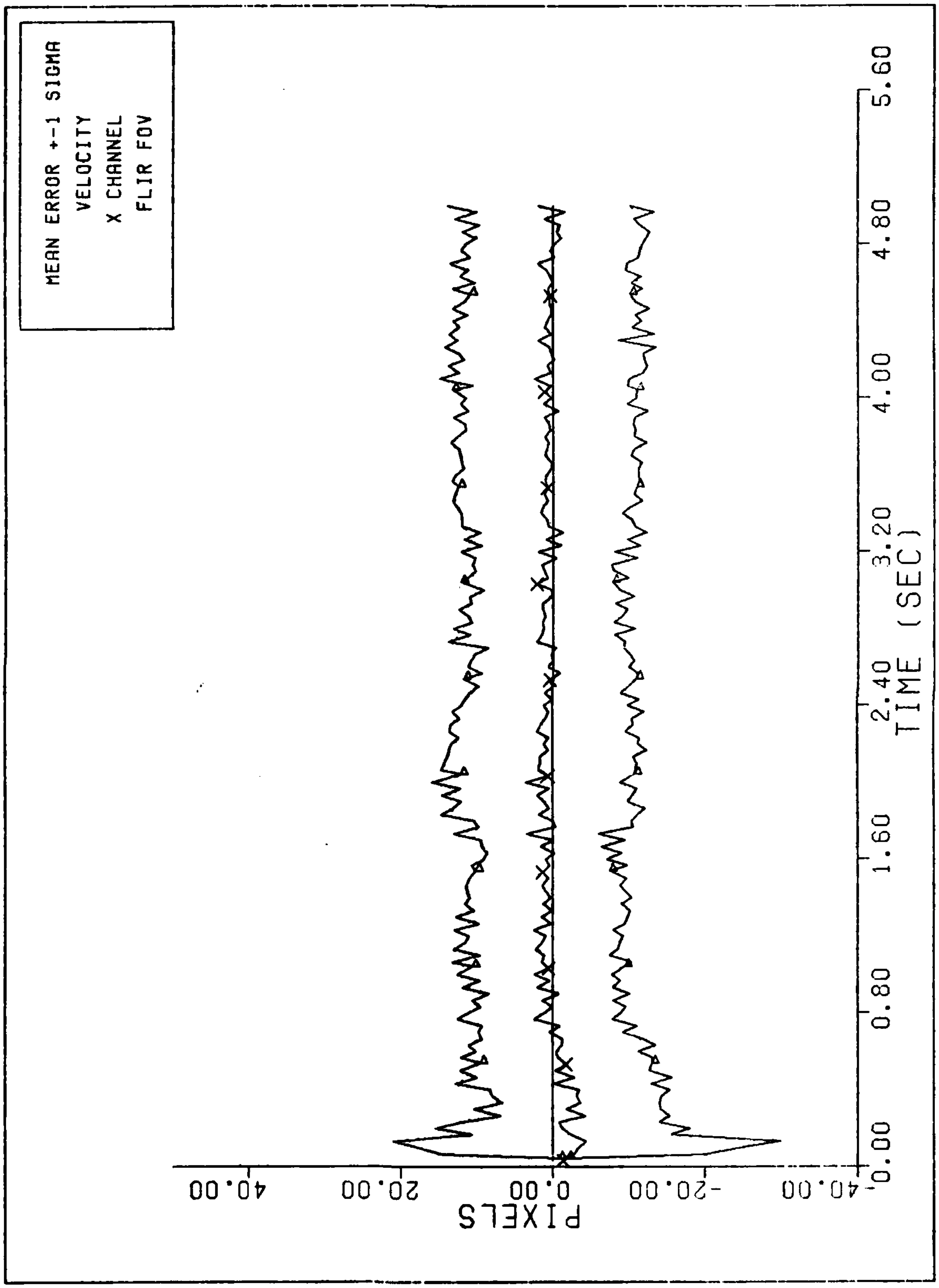
X CHANNEL DYNAMICS ERROR (S/N= 20)

Figure F-9 BM Performance Plot - 3 Spot Image

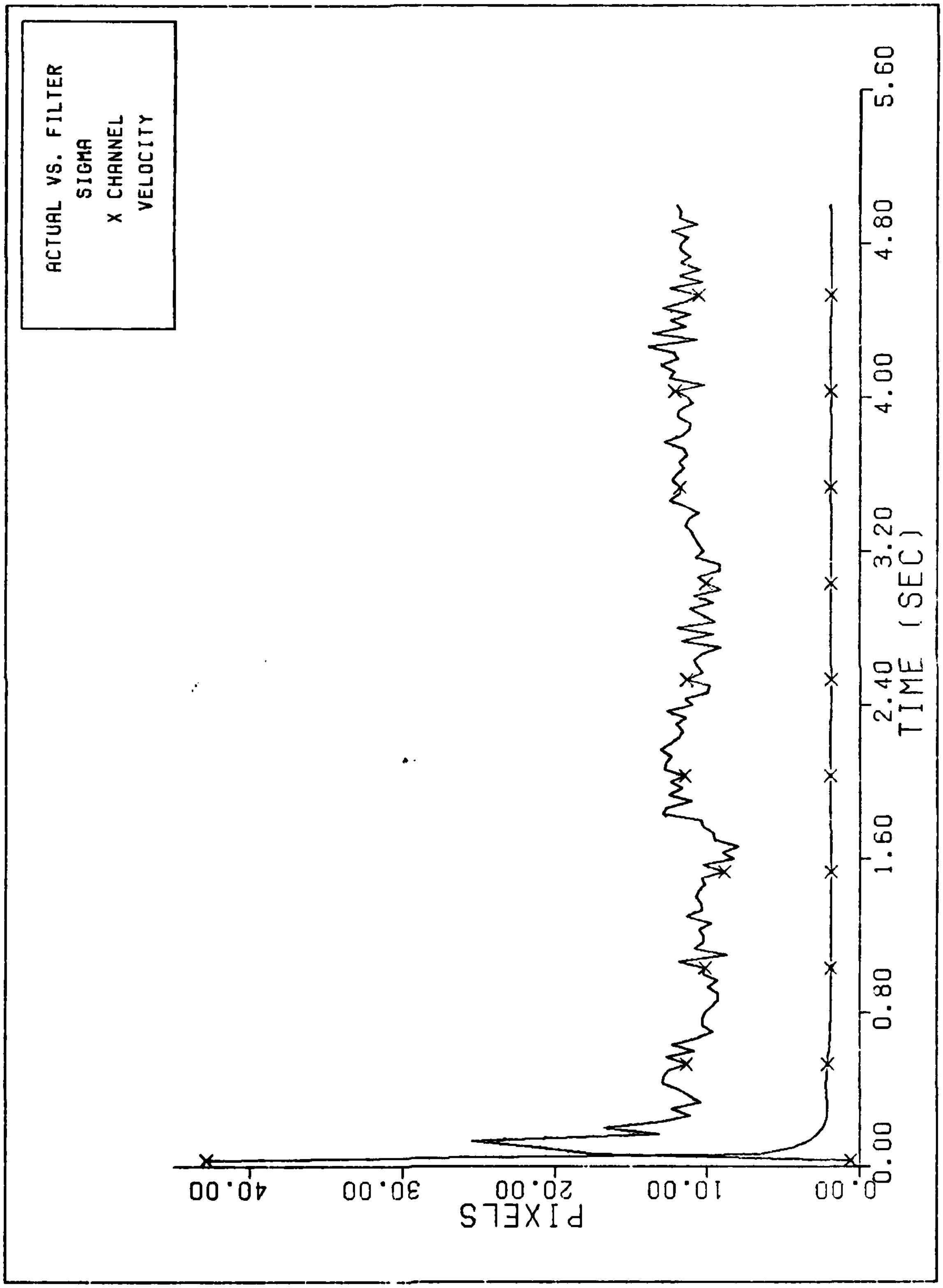


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

Figure F-10 BM Performance Plot - 3 Spot Image

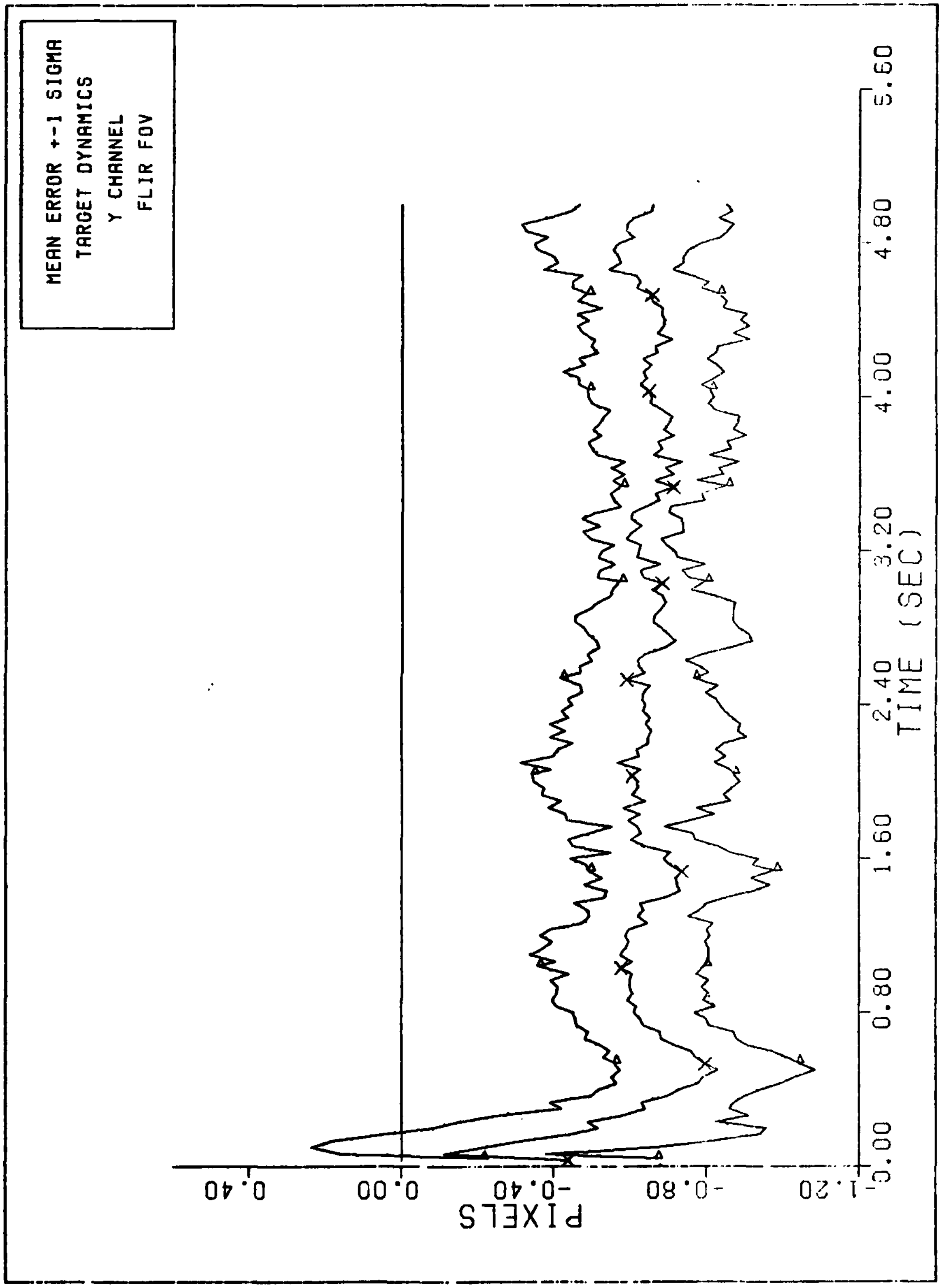


X CHANNEL VELOCITY ERROR (S/N= 20)
Figure F-11 BM Performance Plot - 3 Spot Image



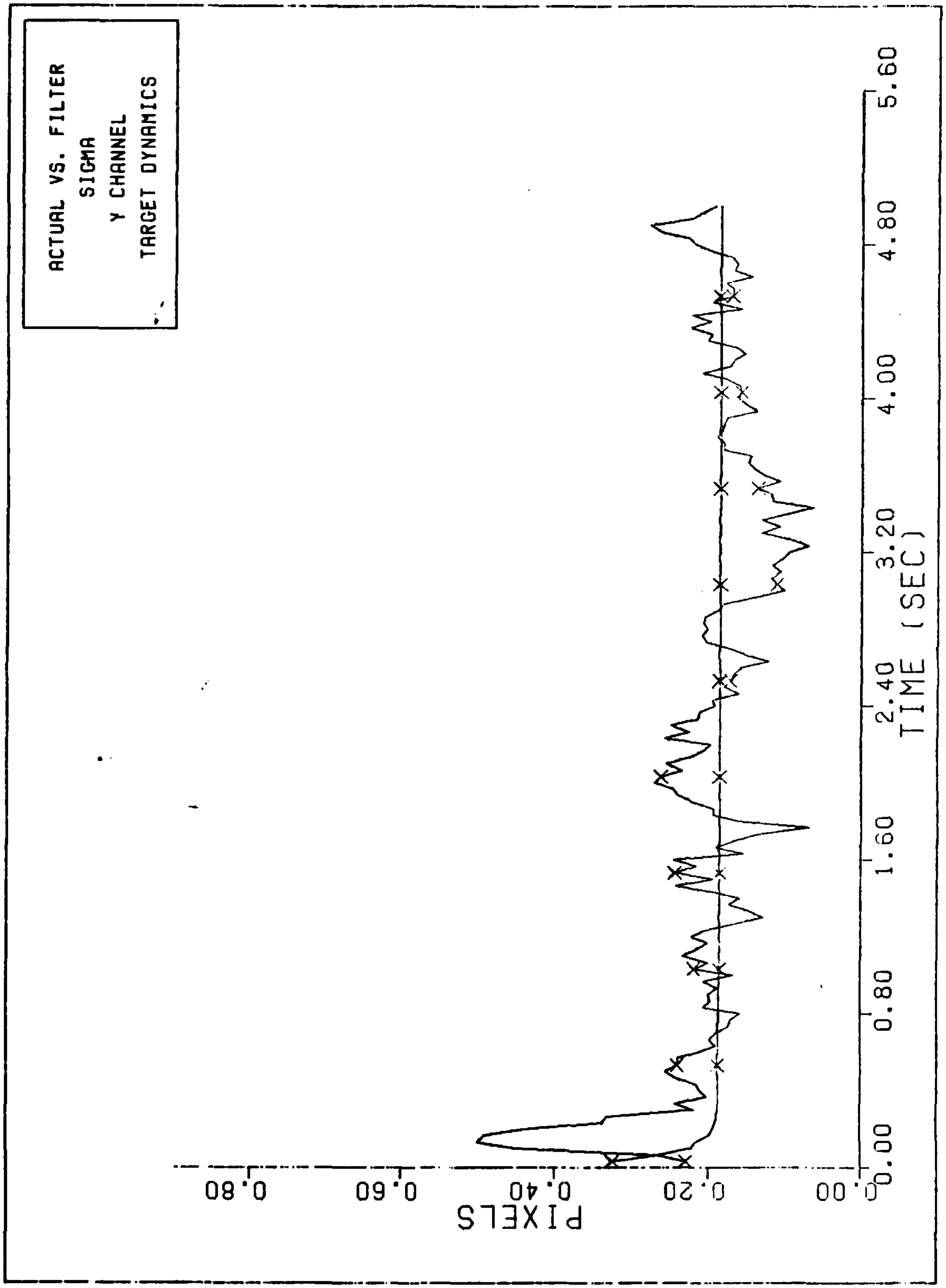
FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

Figure F-12 BM Performance Plot - 3 Spot Image



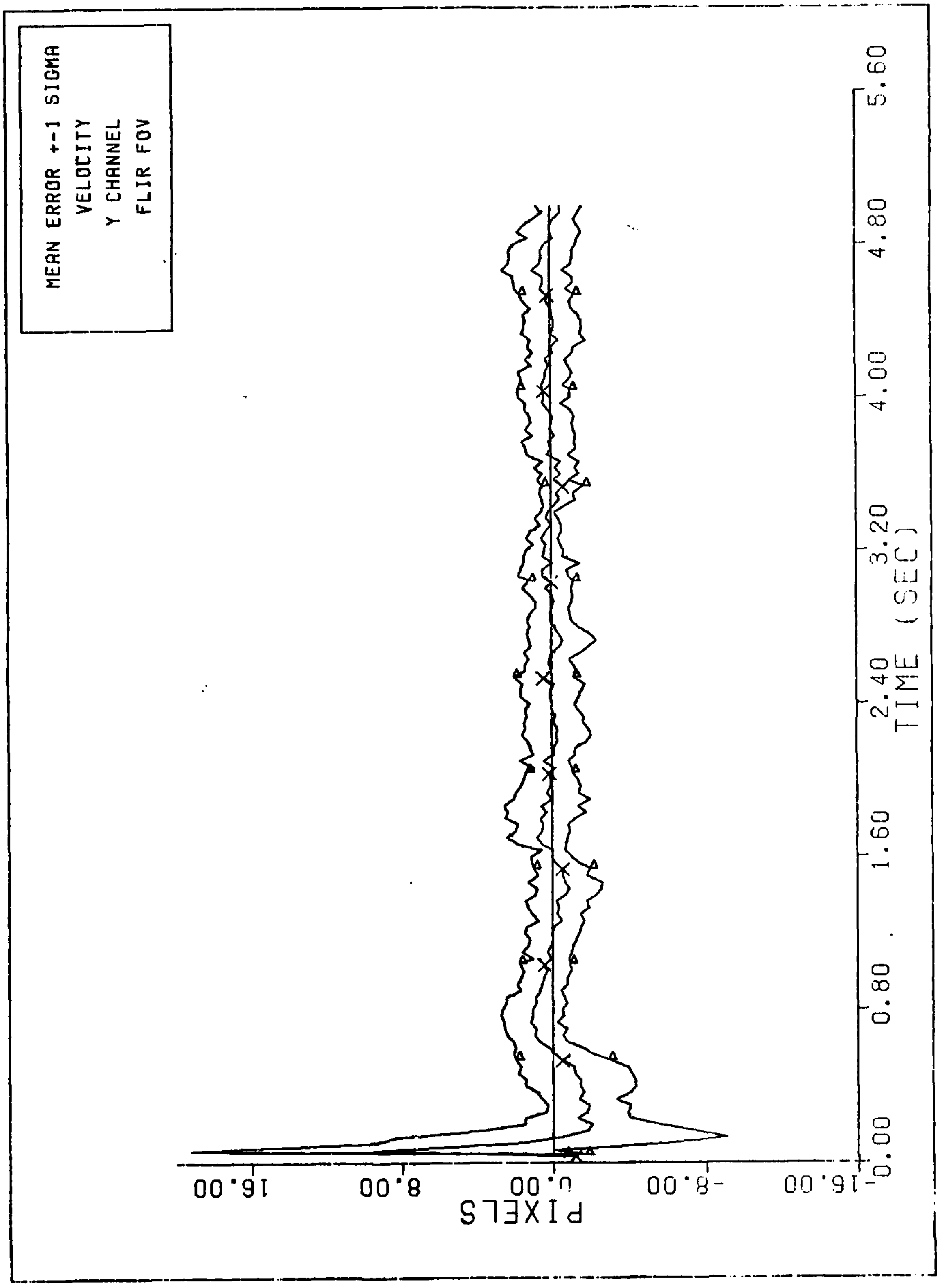
Y CHANNEL DYNAMICS ERROR (S/N=20)

Figure F-13 BM Performance Plot - 3 Spot Image

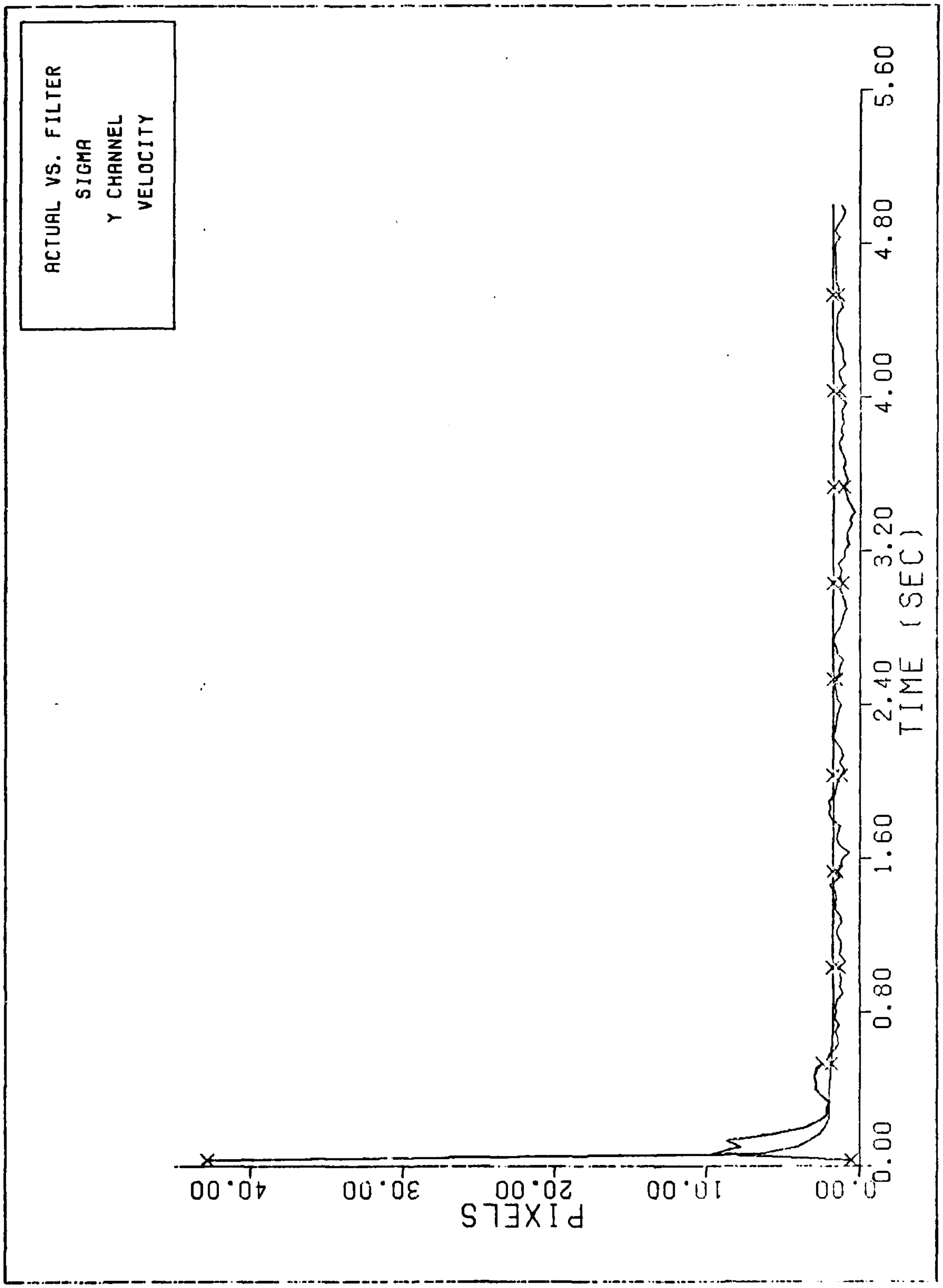


FILTER VS. ACTUAL SIGMA PLOT (S/N =20)

Figure F-14 BM Performance Plot - 3 Spot Image

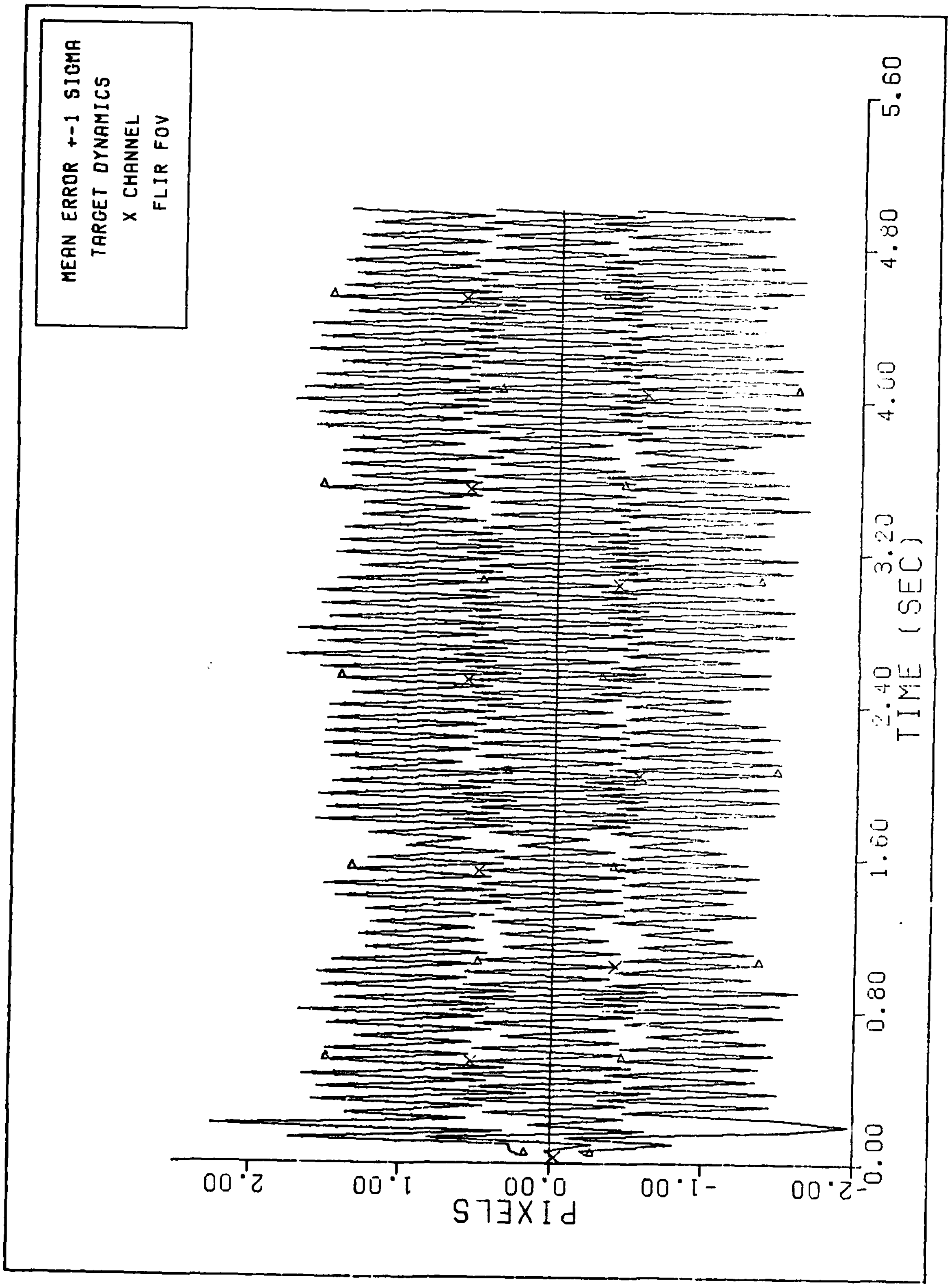


Y CHANNEL VELOCITY ERROR (S/N= 20)
Figure F-15 BM Performance Plot - 3 Spot Image



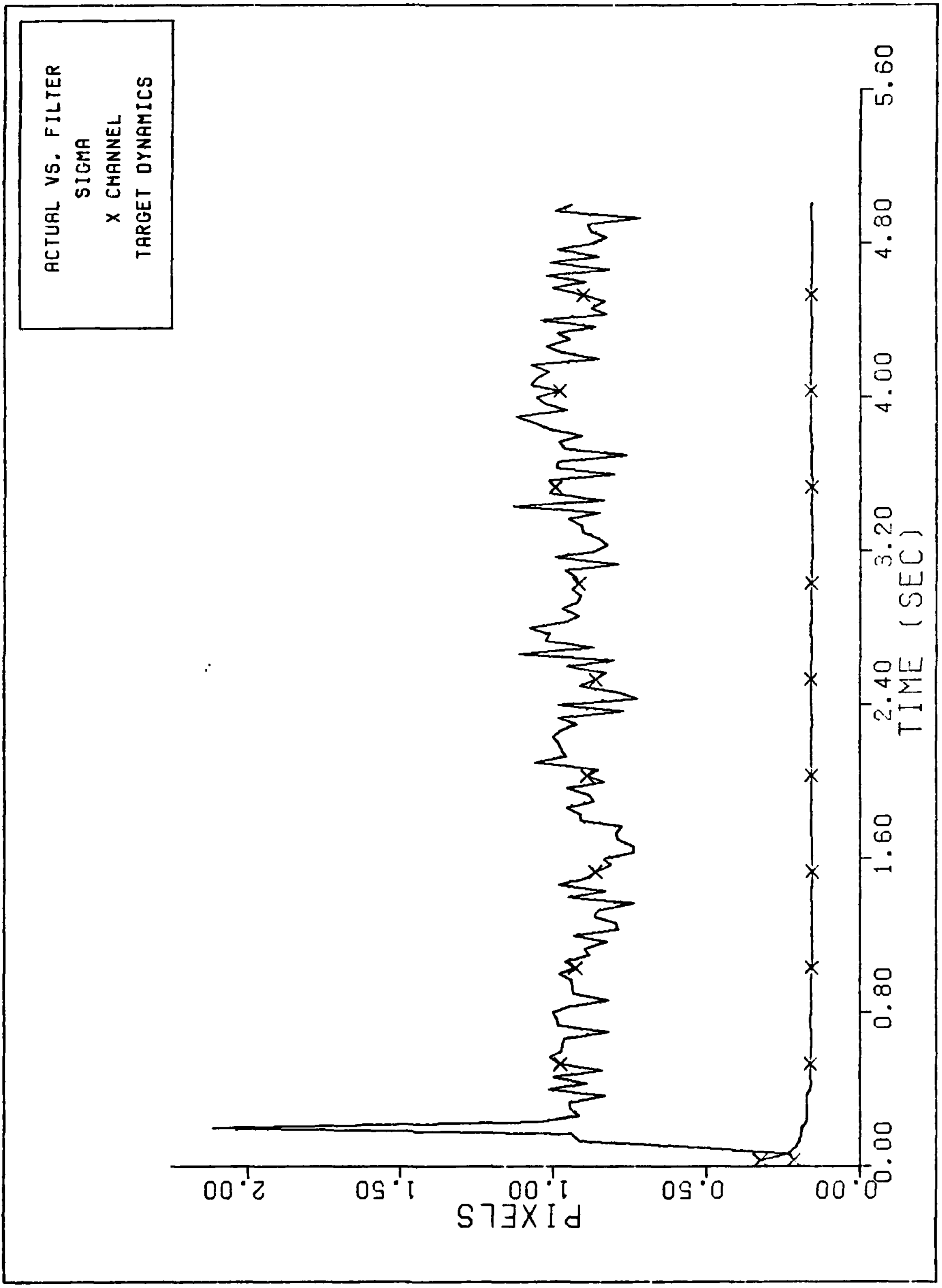
FILTER VS. ACTUAL SIGMA PLOT (S/N =20)

Figure F-16 BM Performance Plot - 3 Spot Image



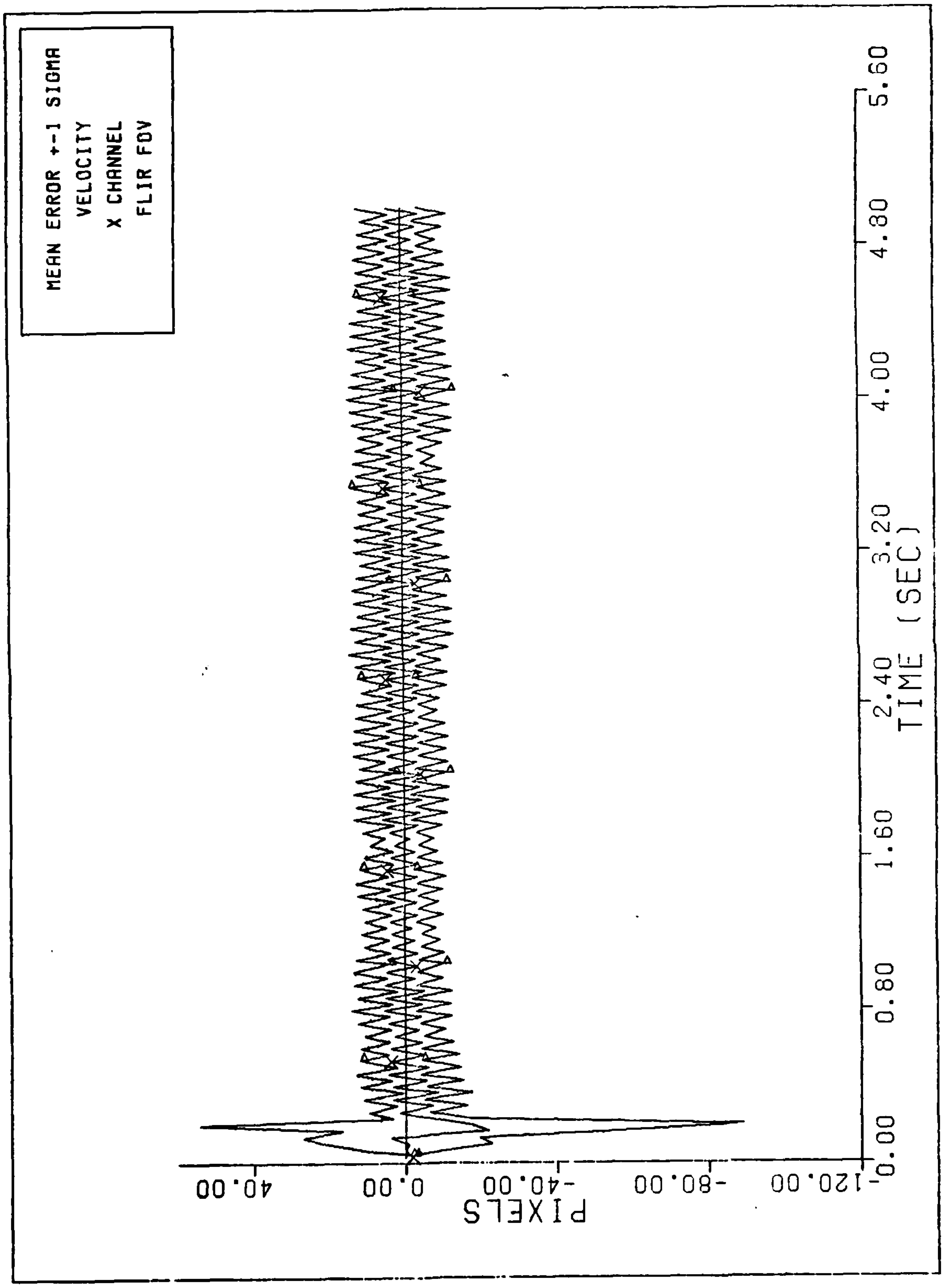
X CHANNEL DYNAMICS ERROR (S/N= 20)

Figure F-17 BM Performance Plot - 3 Spot Image

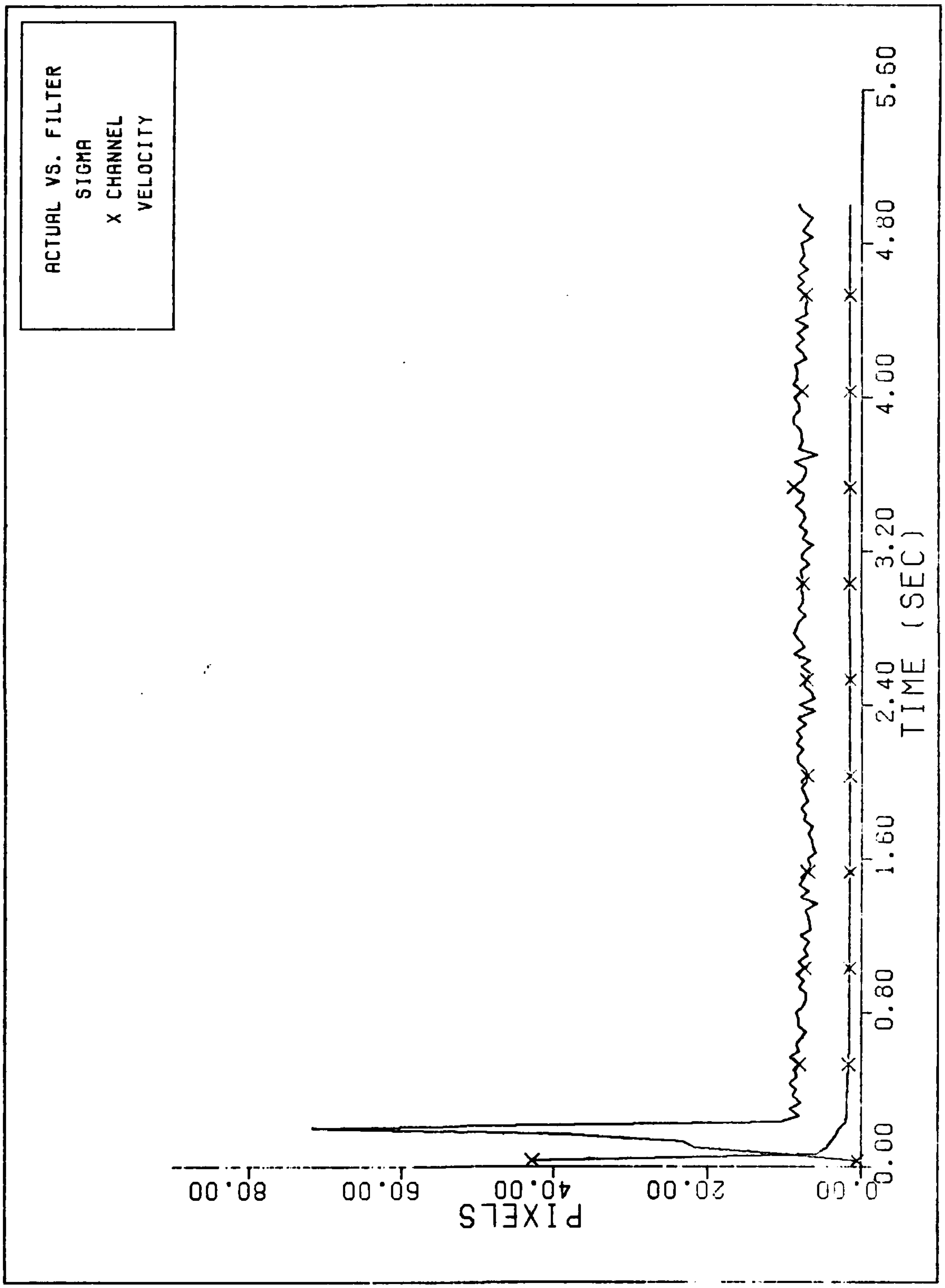


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

Figure F-18 BM Performance Plot - 3 Spot Image

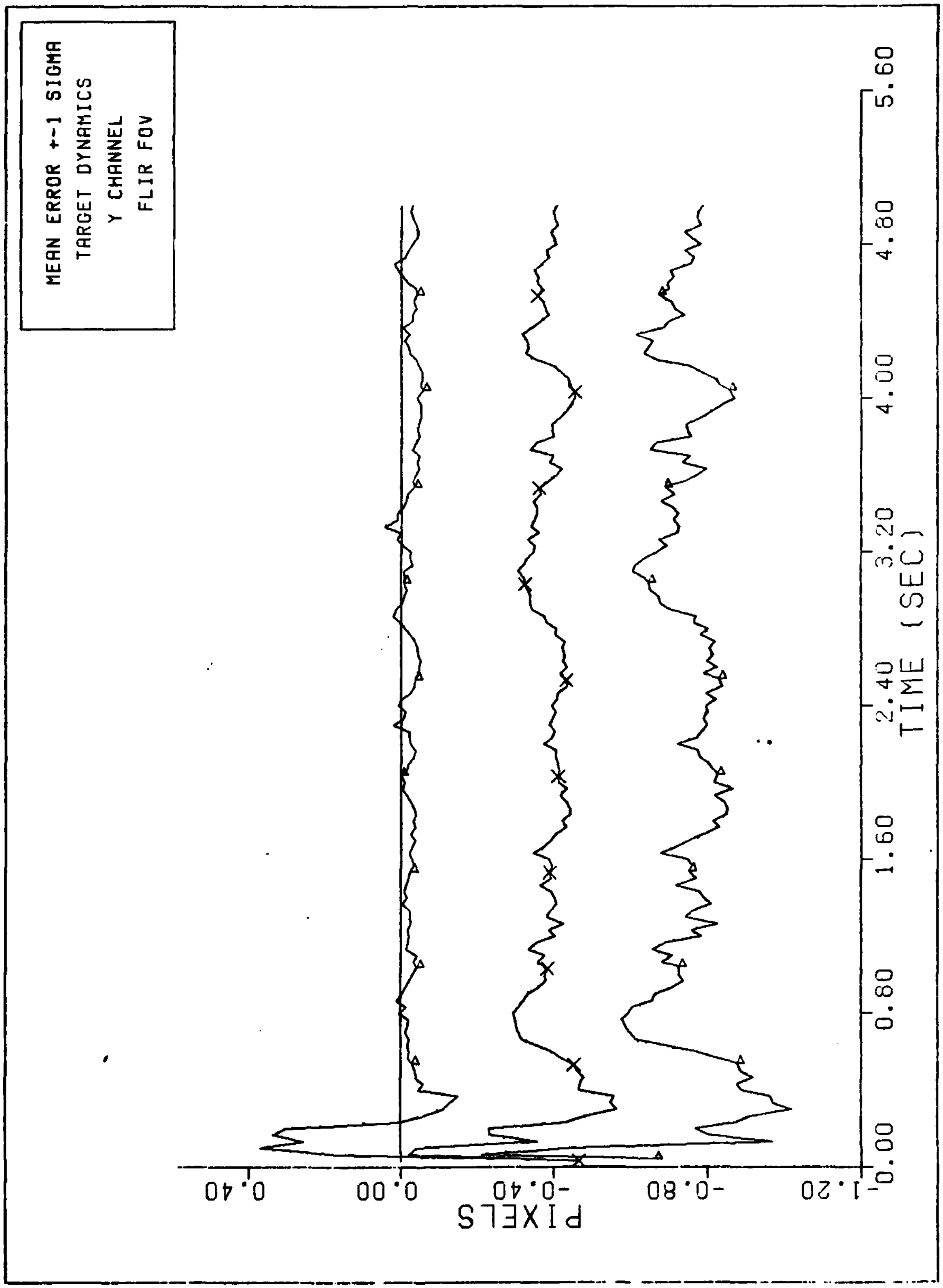


X CHANNEL VELOCITY ERROR (S/N= 20)
Figure F-19 BM Performance Plot - 3 Spot Image

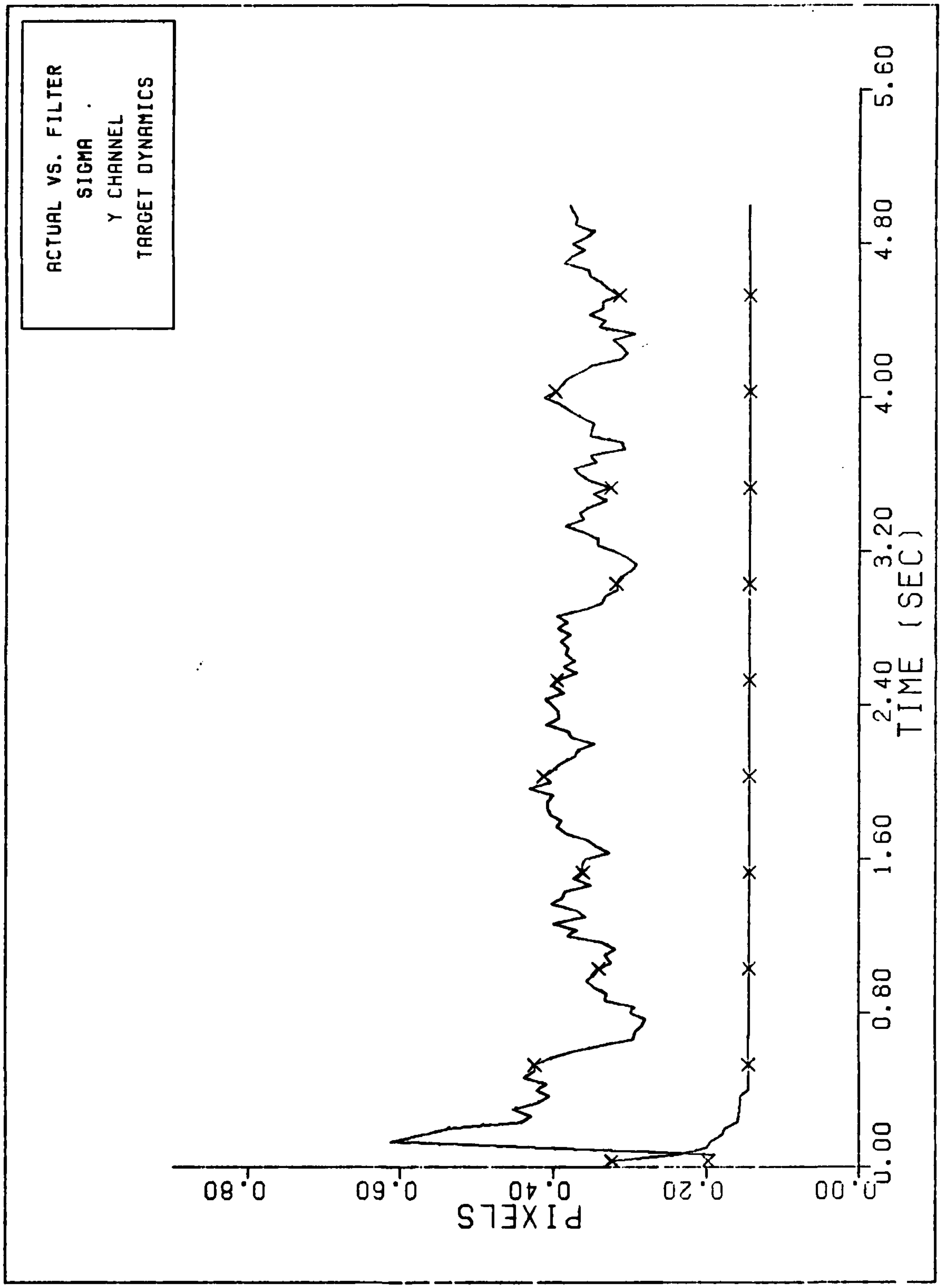


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

Figure F-20 BM Performance Plot - 3 Spot Image

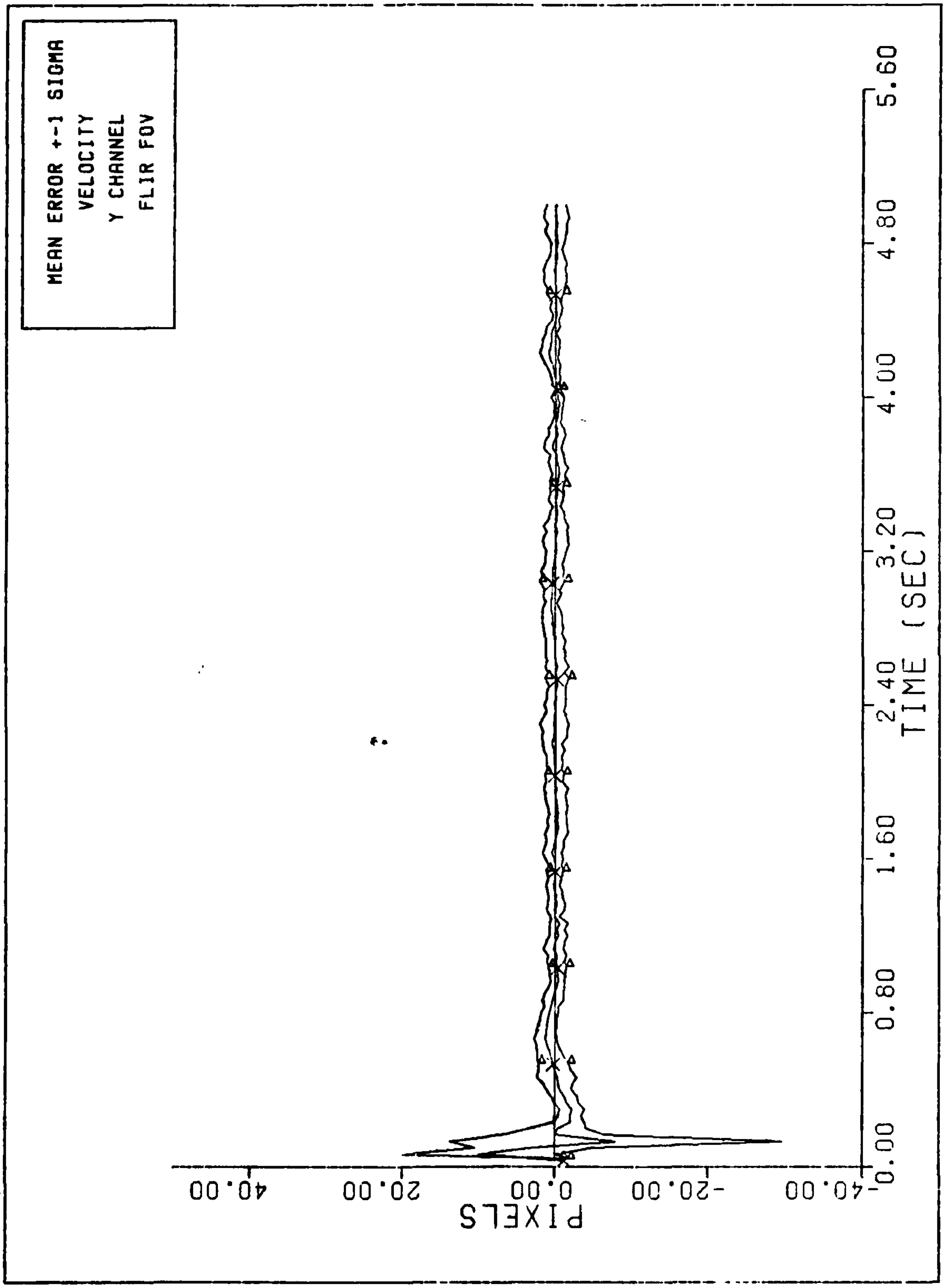


Y CHANNEL DYNAMICS ERROR (S/N=20)
Figure F-21 BM Performance Plot - 3 Spot Image

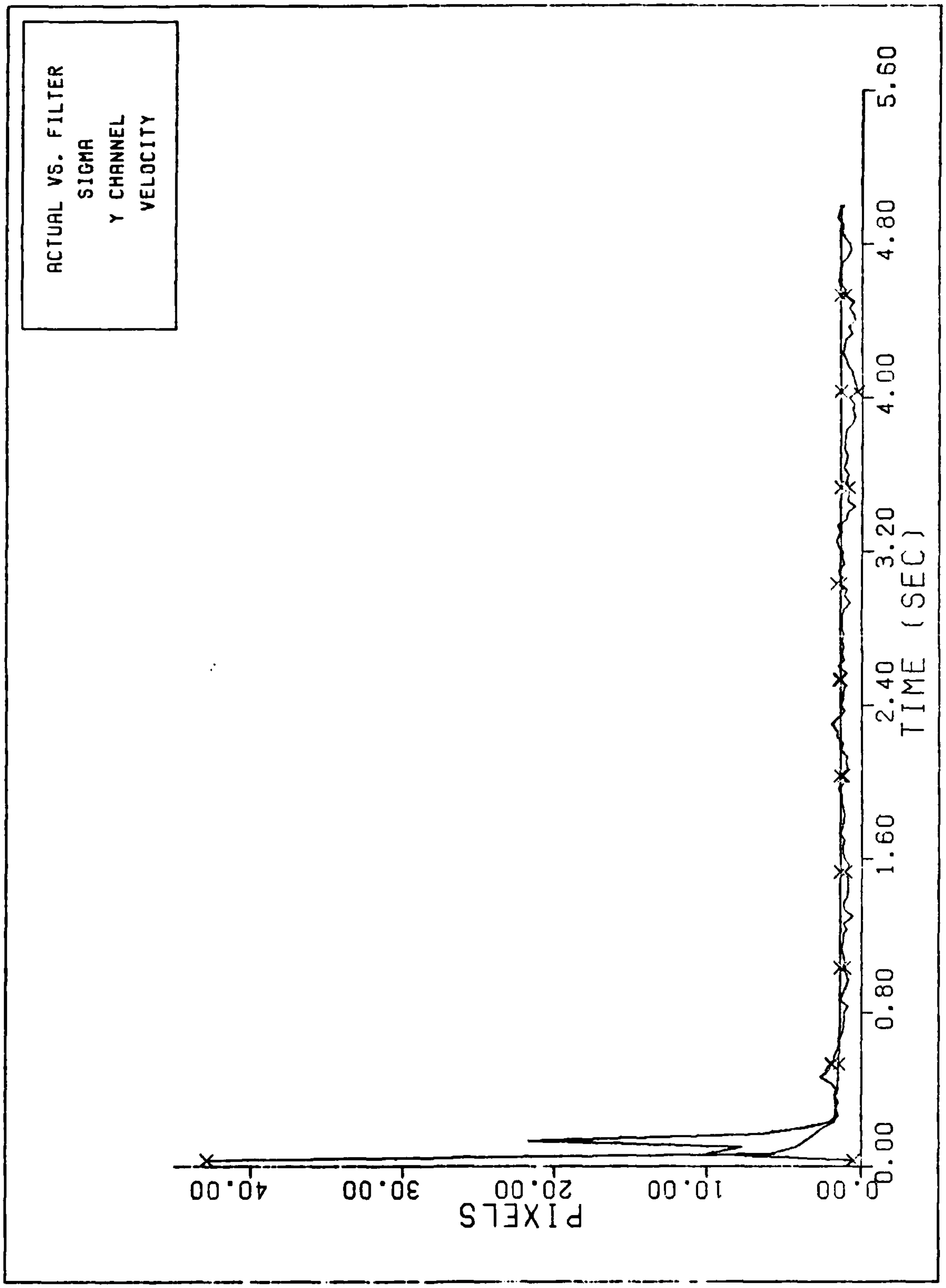


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

Figure F-22 BM Performance Plot - 3 Spot Image

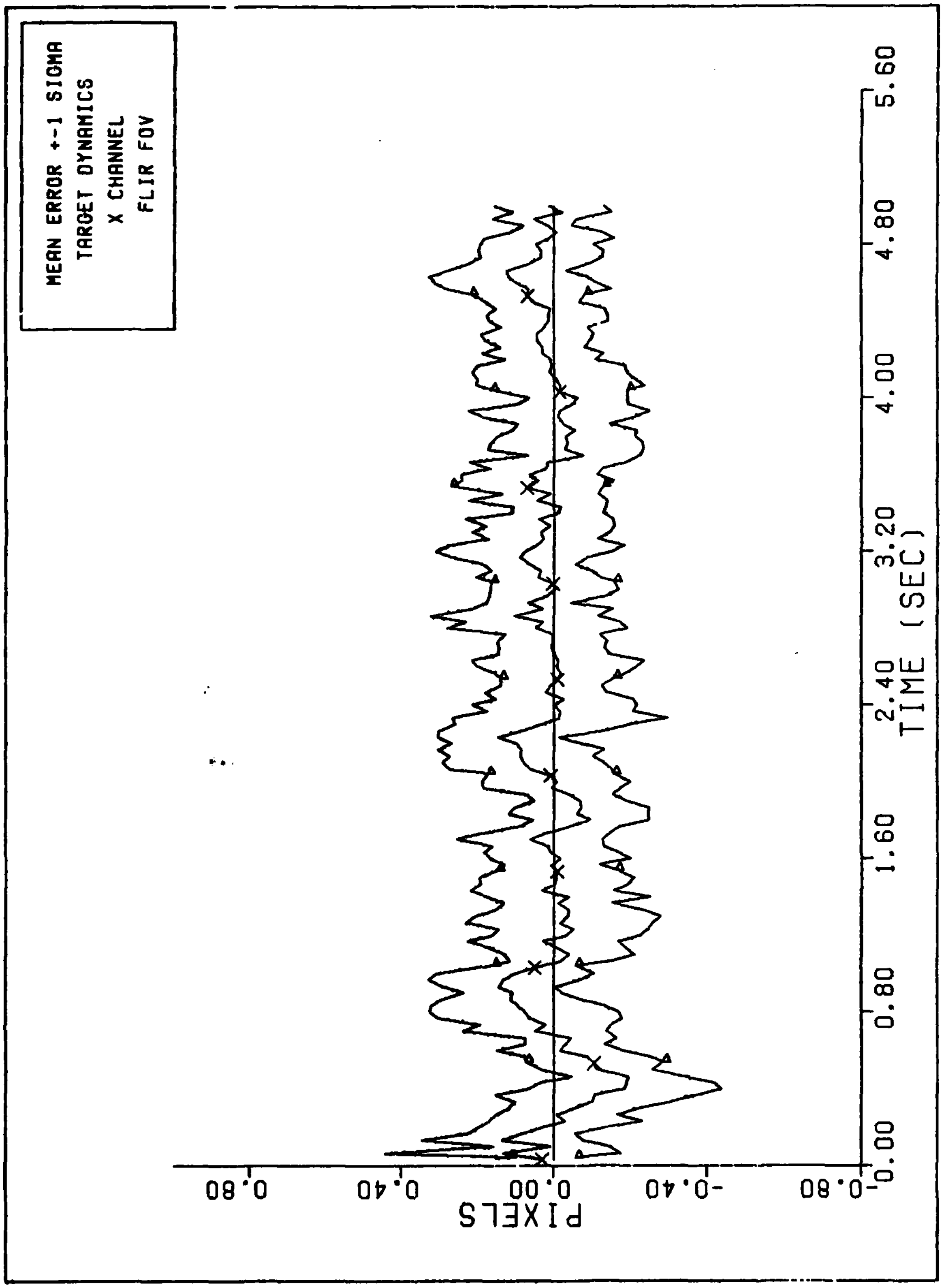


Y CHANNEL VELOCITY ERROR (S/N= 20)
Figure F-23 BM Performance Plot - 3 Spot Image

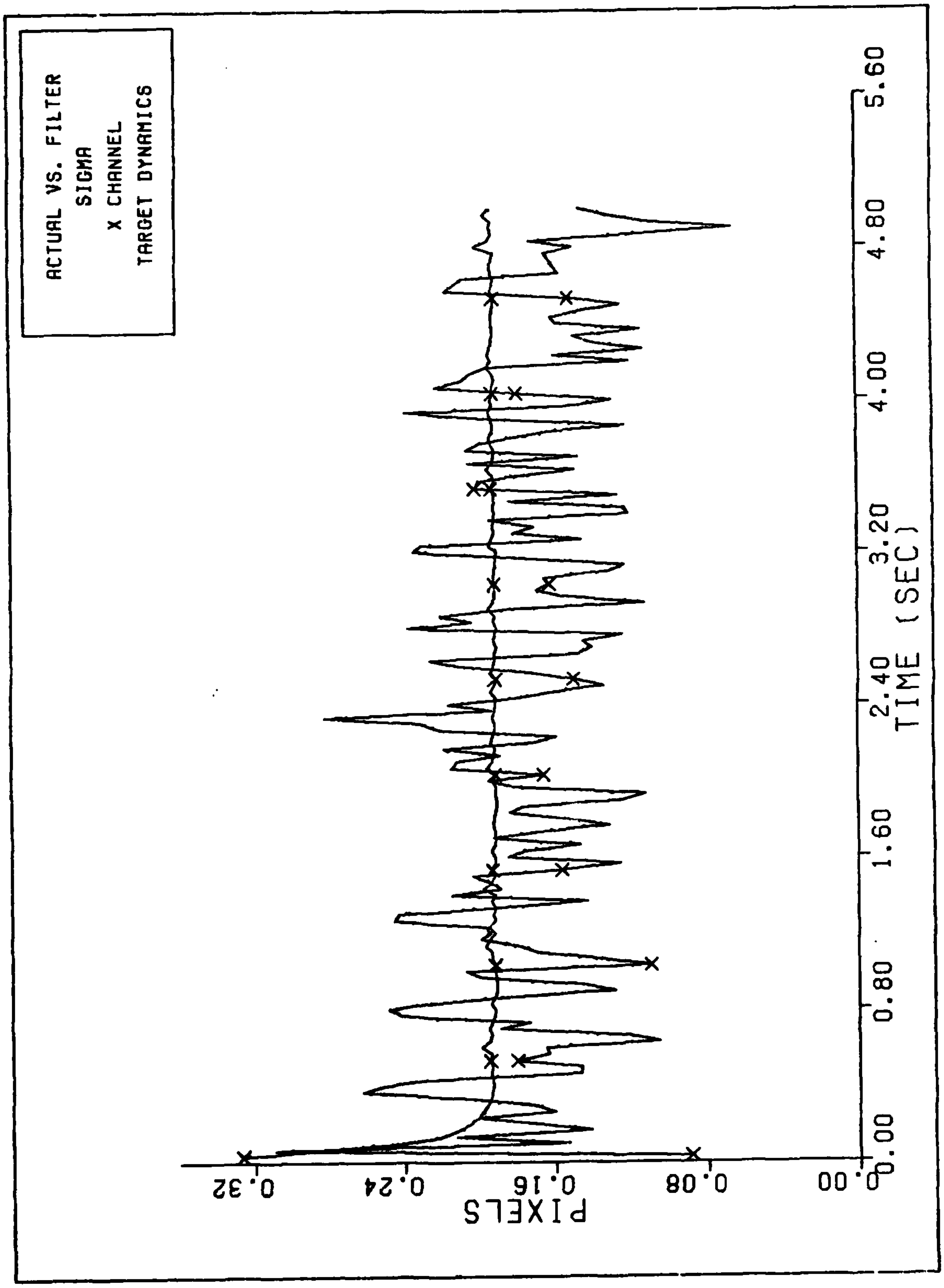


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

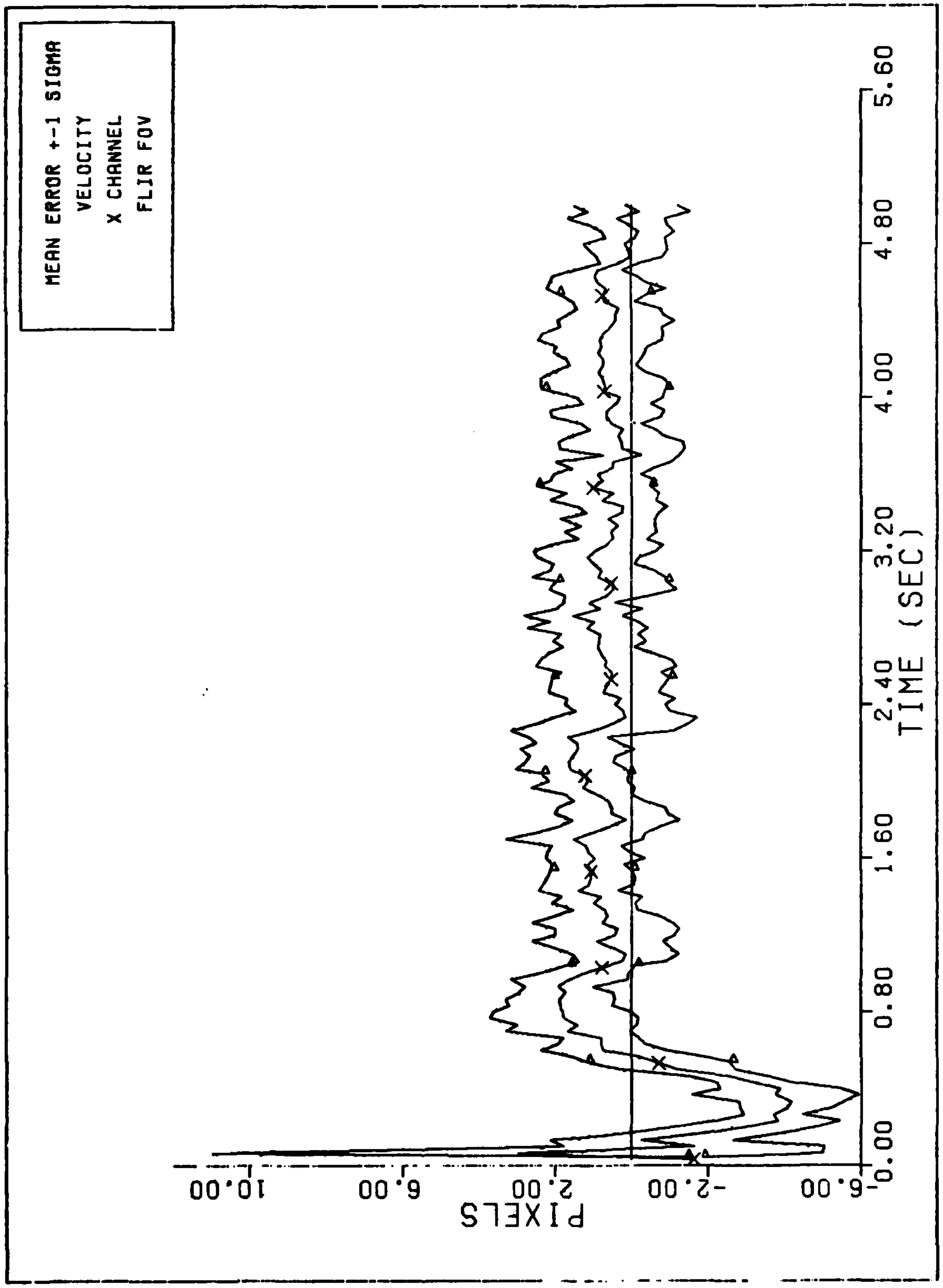
Figure F-24 BM Performance Plot - 3 Spot Image



X CHANNEL DYNAMICS ERROR (S/N= 20)
Figure F-25 BM Performance Plot - 2G pull-up

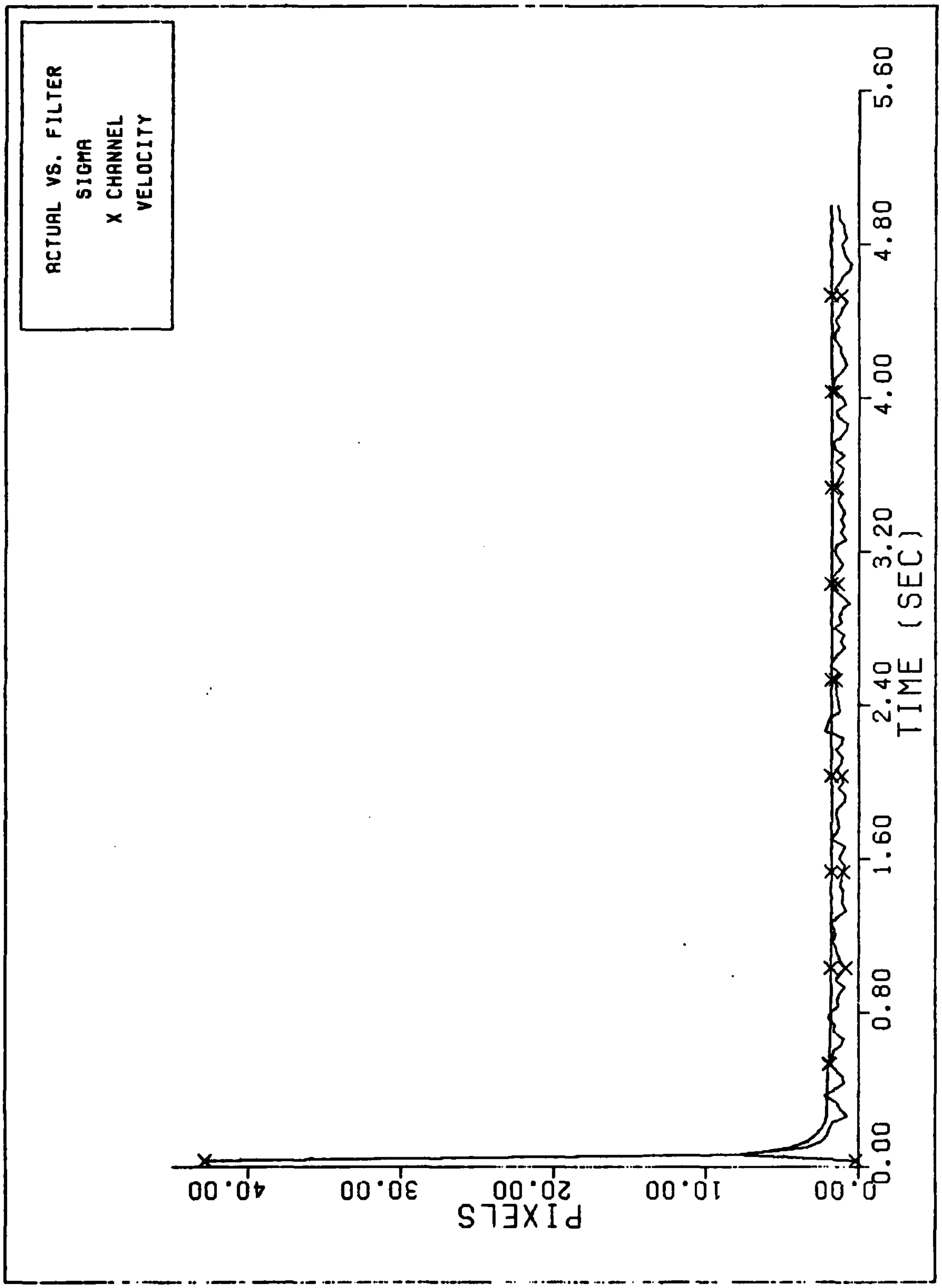


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)
Figure F-26 BM Performance Plot - 2G pull-up



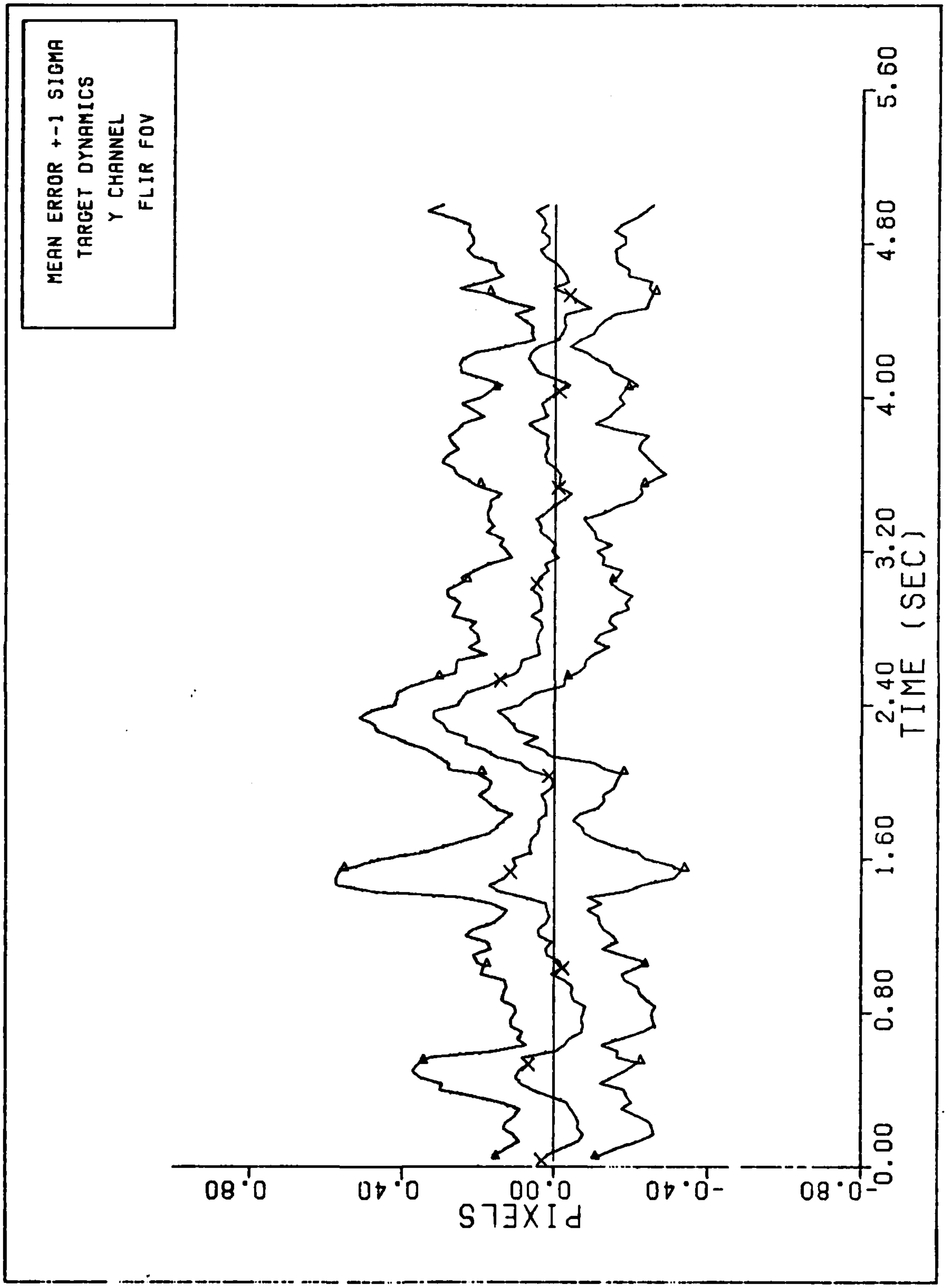
X CHANNEL VELOCITY ERROR (S/N= 20)

Figure F-27 BM Performance Plot - 2G pull-up

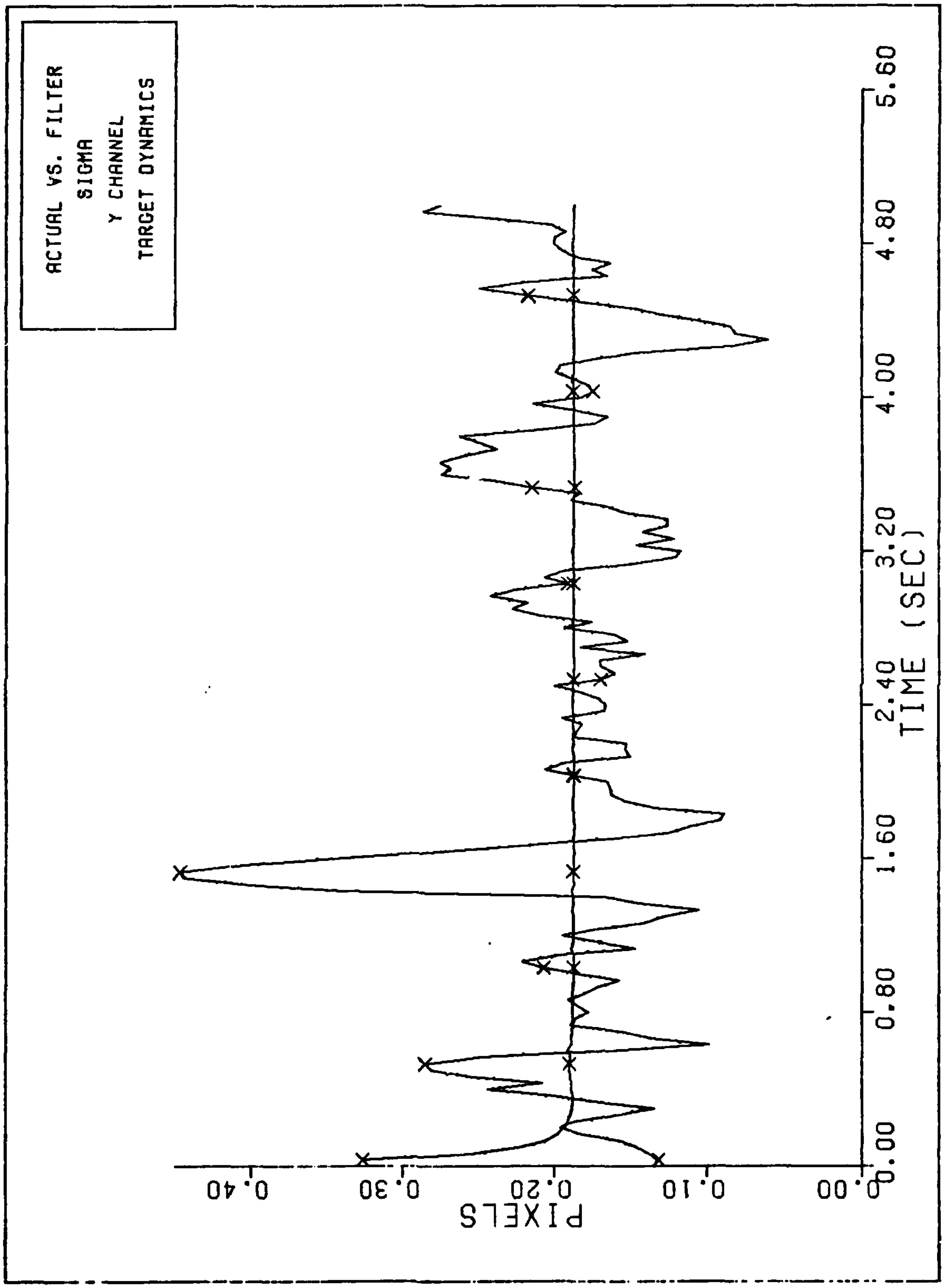


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

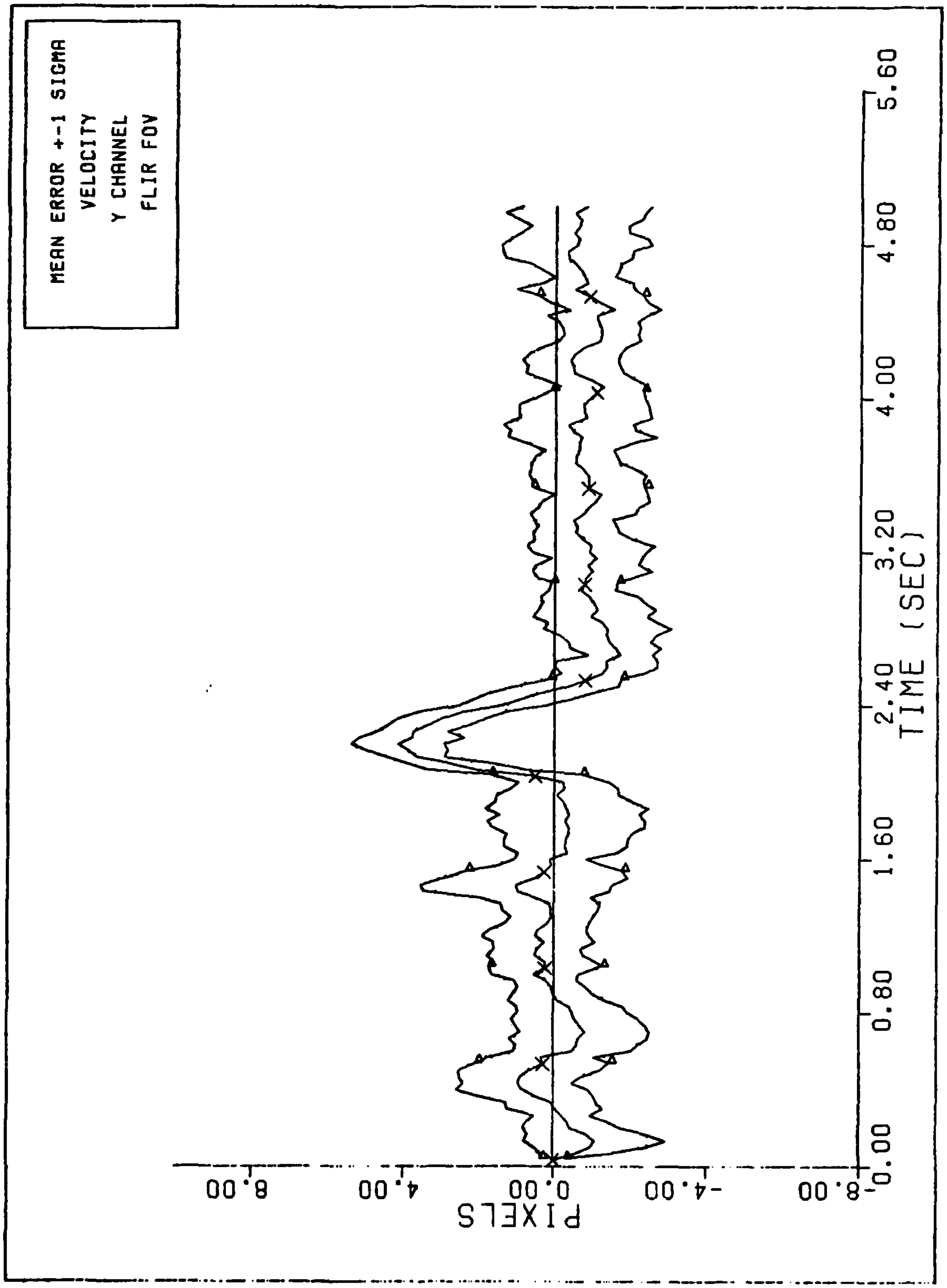
Figure F-28 BM Performance Plot - 2G pull-up



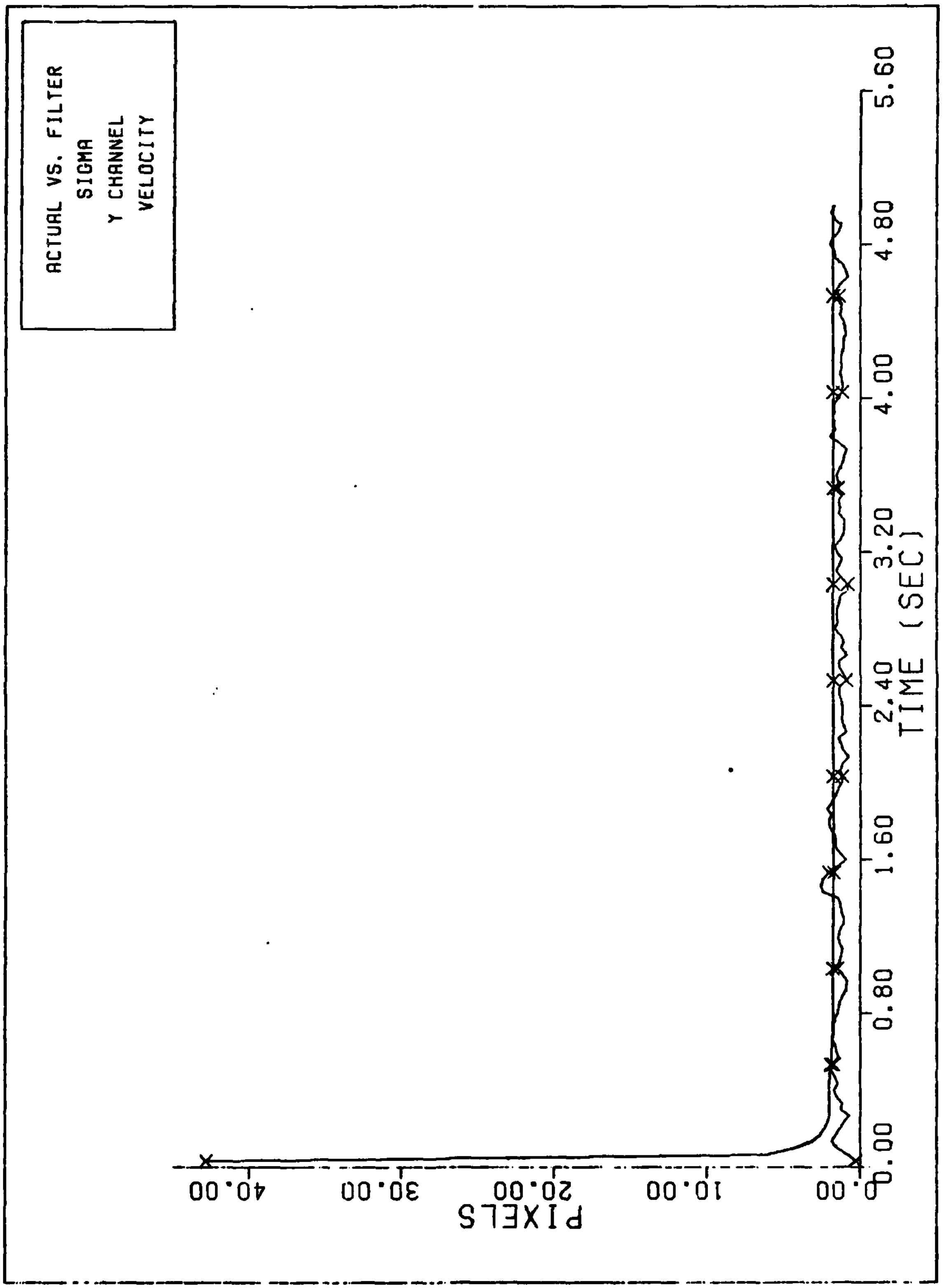
Y CHANNEL DYNAMICS ERROR (S/N= 20)
Figure F-29 BM Performance Plot - 2G pull-up



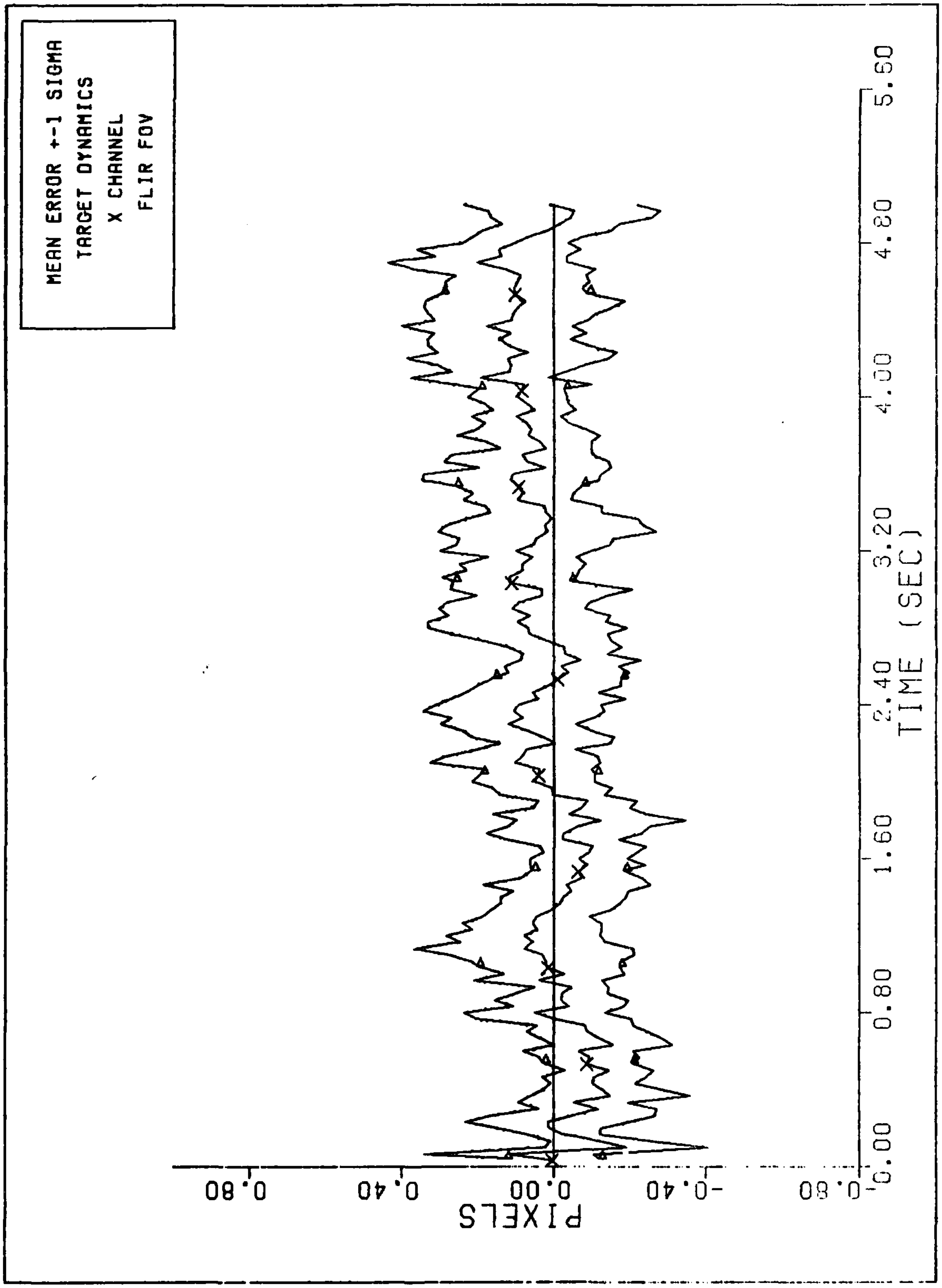
FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)
Figure F-30 BM Performance Plot - 2G pull-up



Y CHANNEL VELOCITY ERROR (S/N= 20)
Figure F-31 BM Performance Plot - 2G pull-up

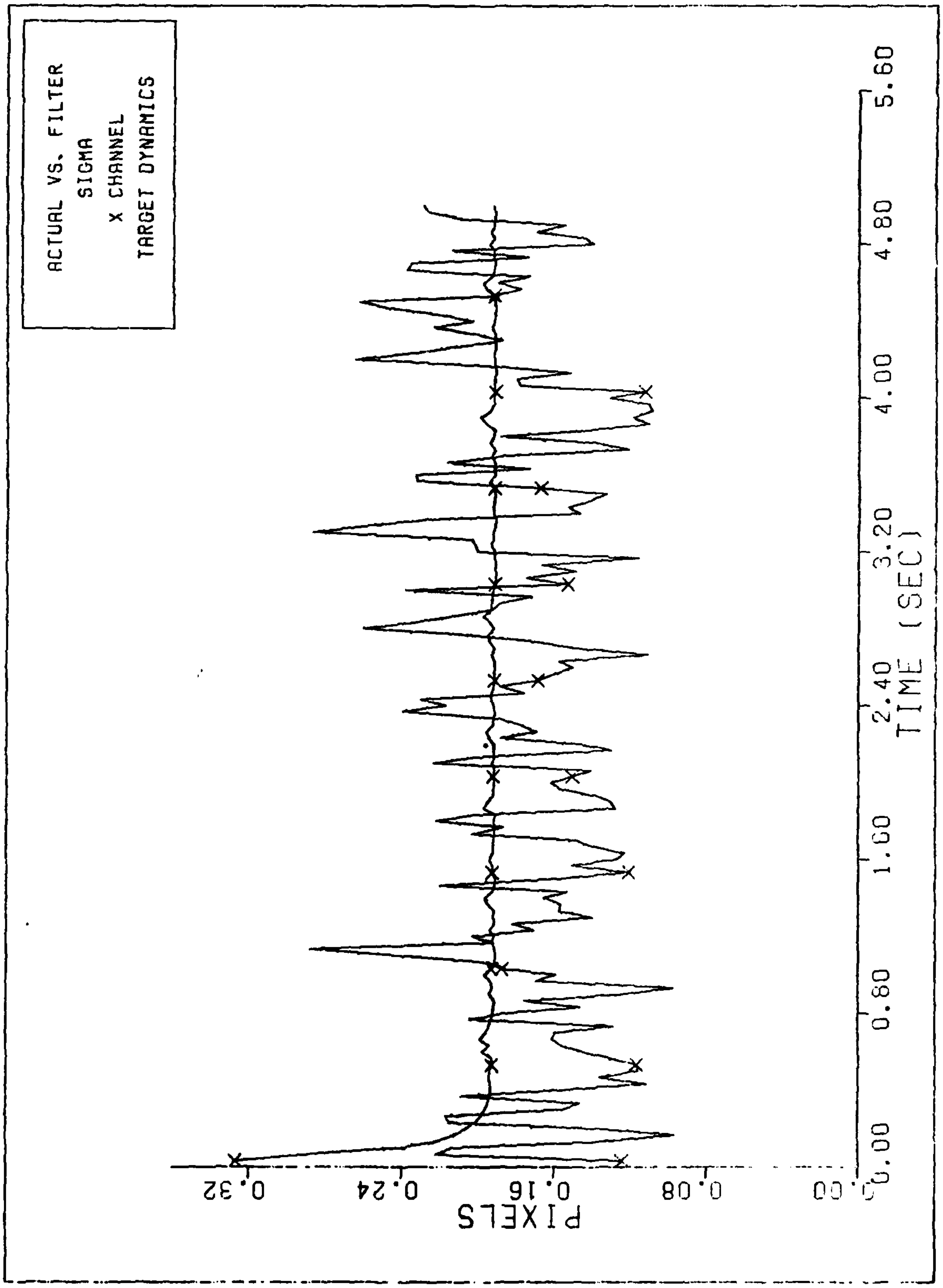


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)
Figure F-32 BM Performanne Plot - 2G pull-up

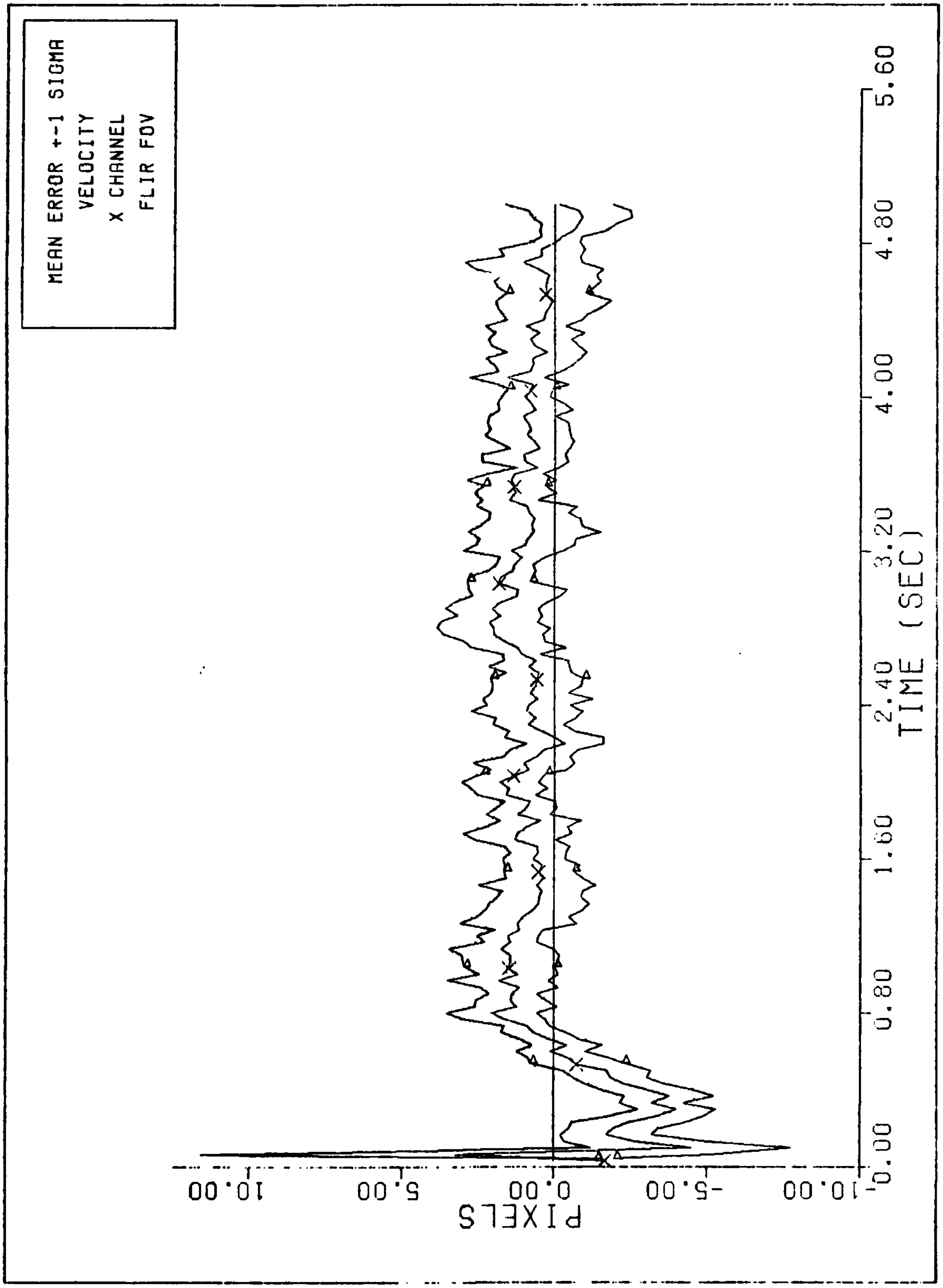


X CHANNEL DYNAMICS ERROR (S/N= 20)

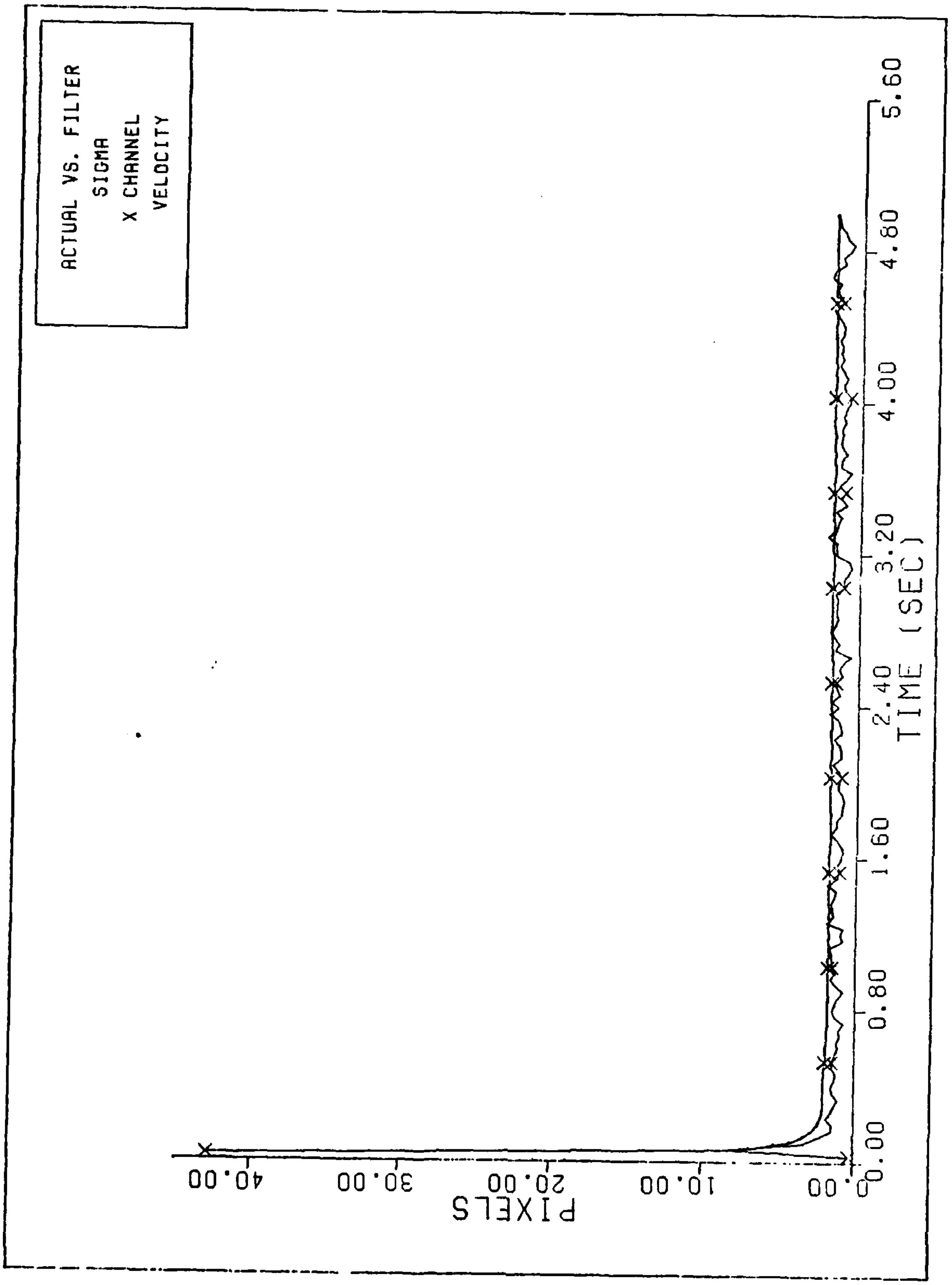
Figure F-33 BM Performance Plot - 5G turn



FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)
Figure F-34 BM Performance Plot - 5G turn

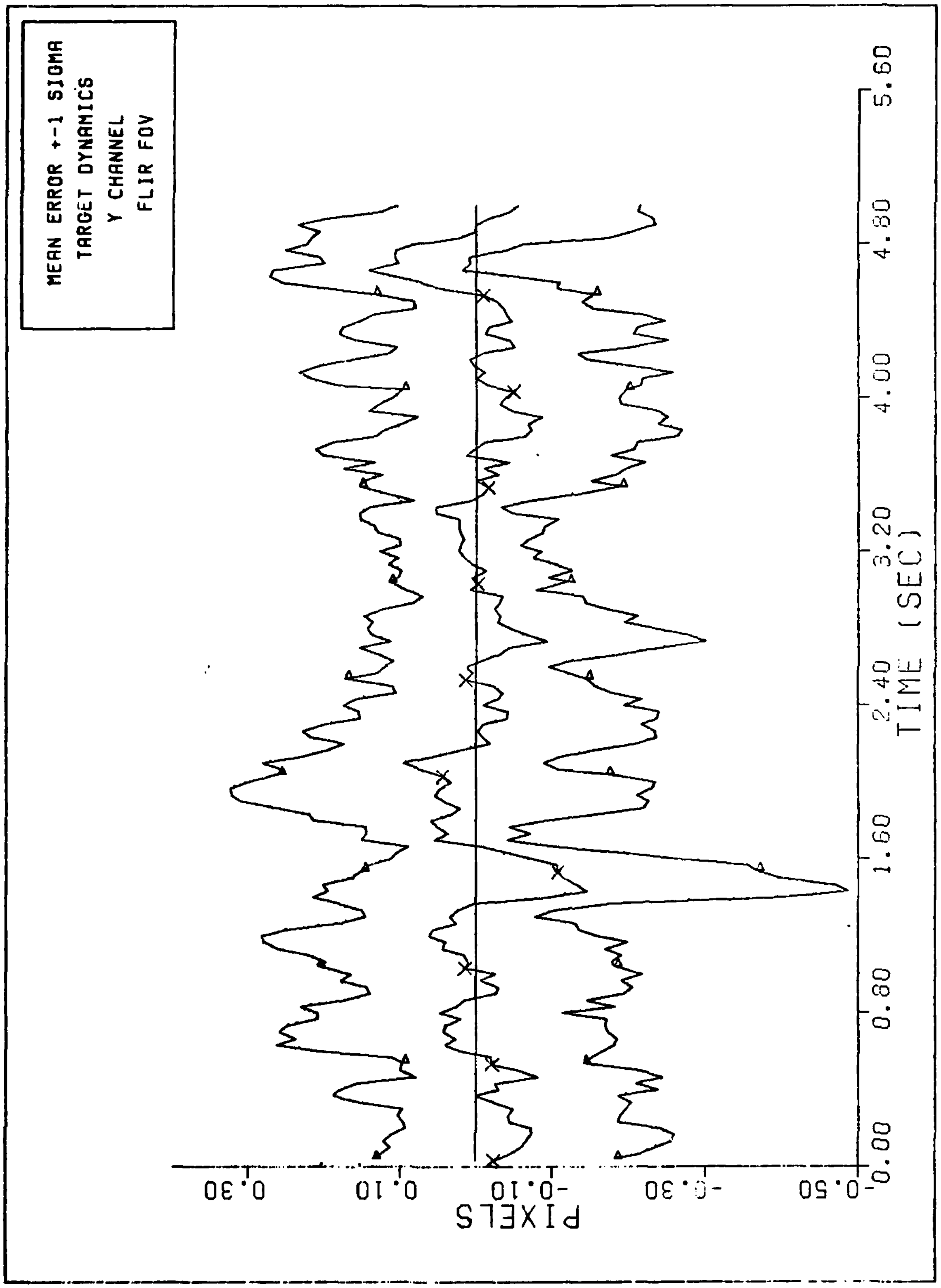


X CHANNEL VELOCITY ERROR (S/N= 20)
Figure F-35 BM Performance Plot - 5G turn

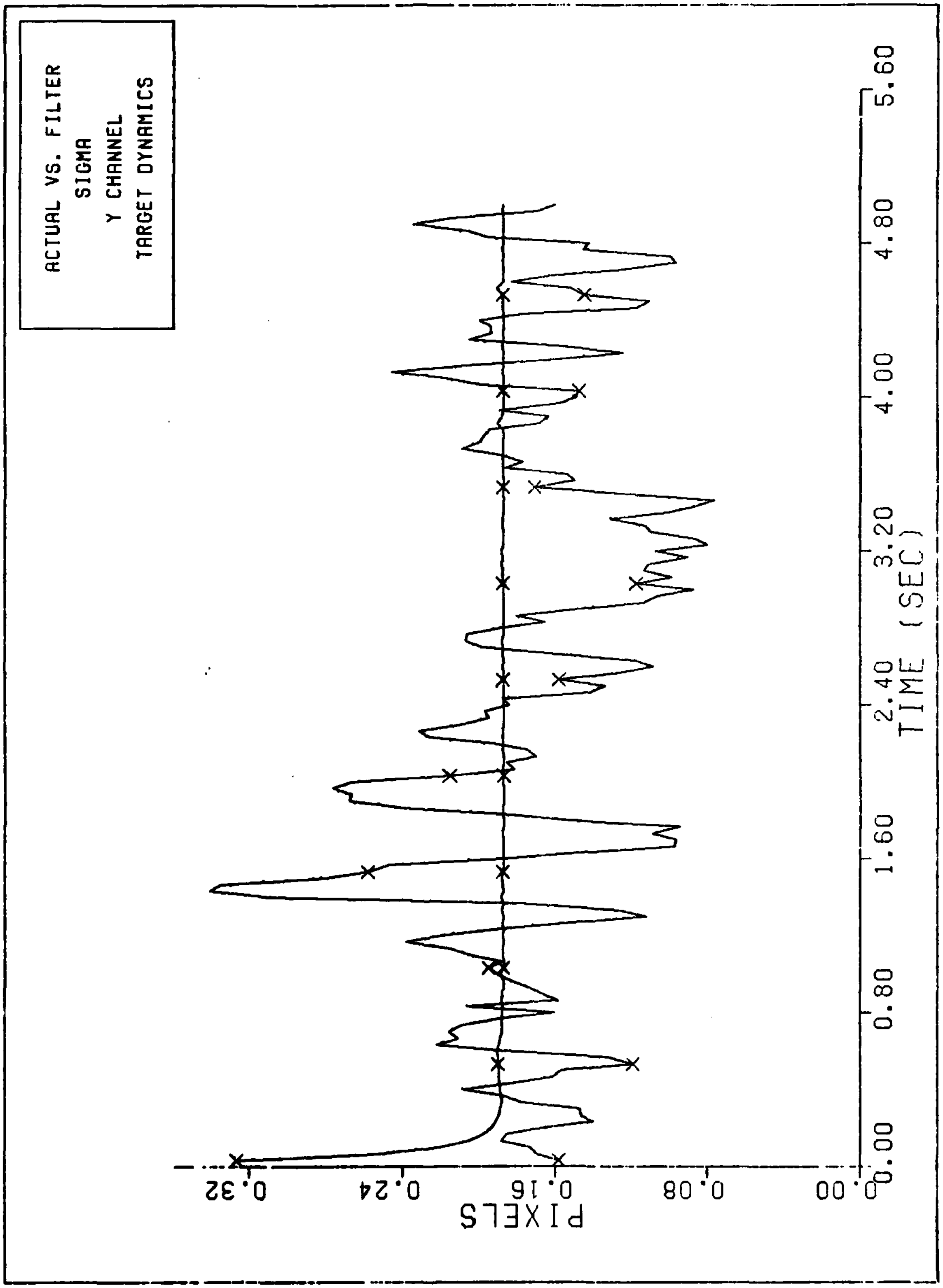


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

Figure F-36 BM Performance Plot - 5G turn

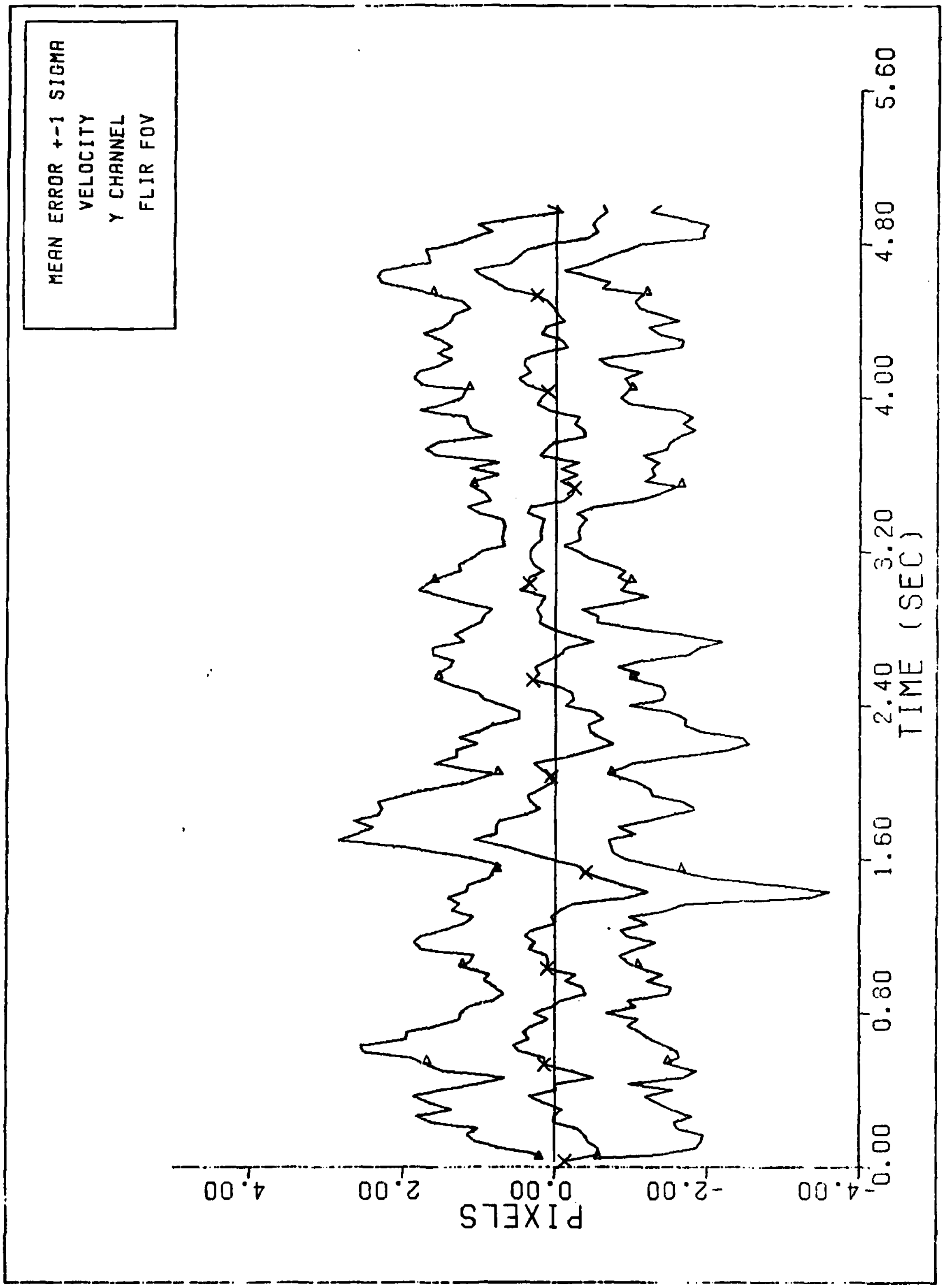


Y CHANNEL DYNAMICS ERROR (S/N=20)
Figure F-37 BM Performance Plot - 5G turn



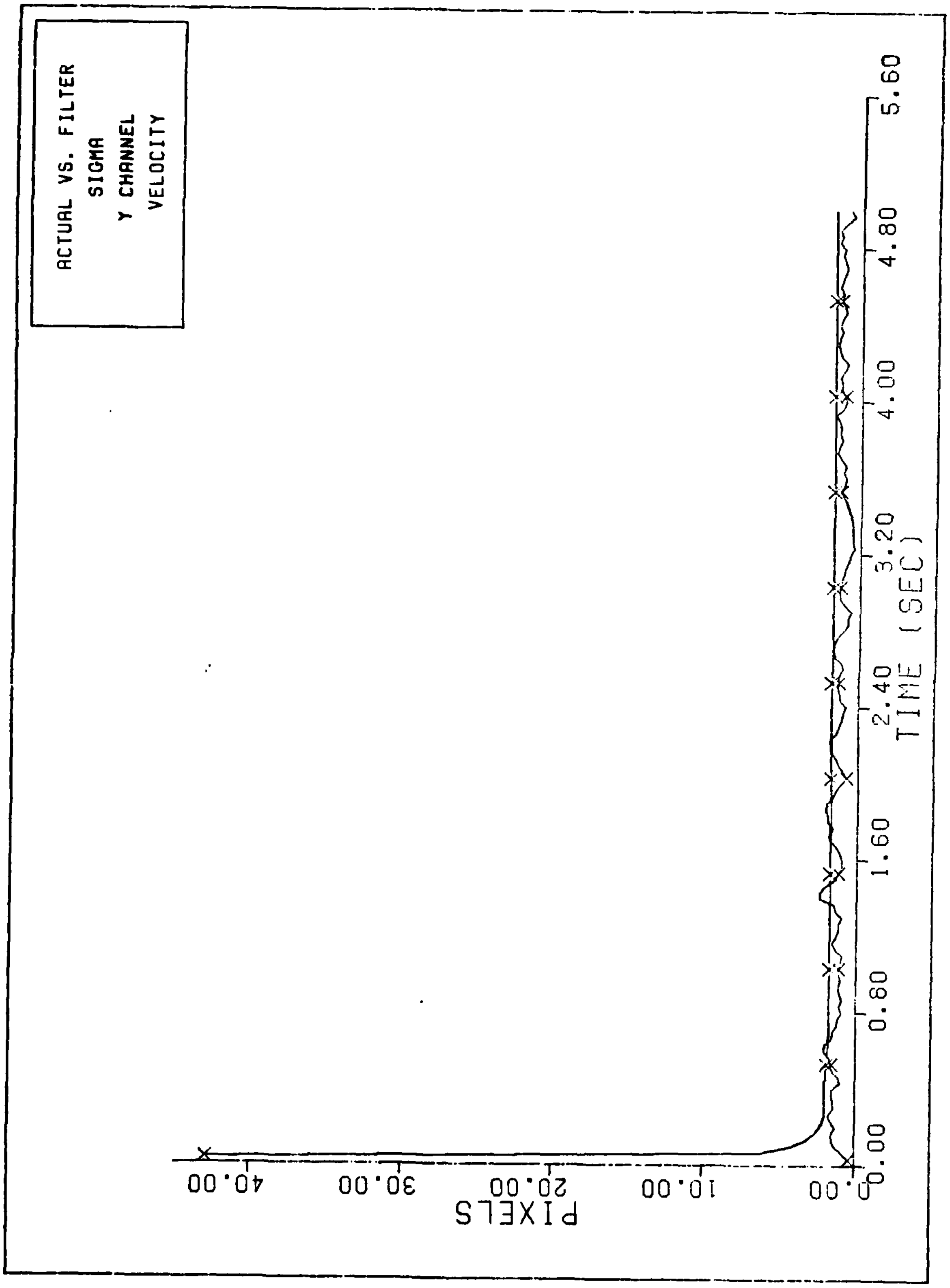
FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

Figure F-38 BM Performance Plot - 5G turn



Y CHANNEL VELOCITY ERROR (S/N= 20)

Figure F-39 BM Performance Plot - 5G turn



FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

Figure F-40 BM Performance Plot - 5G turn

Appendix G

Computer Software

This appendix contains the Fortran source code for the comuter programs used in this study. The first portion of the appendix contains the complete implementation of the algorithm of Figure 1 with the first order Gauss-Markov Filter subroutines. This is followed by the corresponding subroutines that must be substituted to implement the constant turn-rate filter. Finally, the plot routine used to generate the plots for this research is presented. This routine is designed for on-line use with the HP plotter.

Main Program - GM Filter

```
      PROGRAM MAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE8,
     1 DEBUG=OUTPUT)
      REAL EVO(3),EPVO(3),EPPVO(3),EV(3),EPV(3),EPPV(3),RF(2,3)
      REAL SIGV(3),SIGPV(3),SIGVO(3),SIGPVO(3),DELV(3),DELPV(3)
      REAL IMAX(3),S(12),XMAX(3),YMAX(3),R(64,64),RINV(64,64)
      REAL LINH(64,8),NLINH(64)
      REAL XT(8,1),PHIT(8,8),BD(8,4),UT(4),QDROOT(8,8),H(2,8),YT(2,1
      REAL W(64),V(64),UC(576)
      REAL IC(3,3),RROOT(64,64),RFIL(64,64)
      REAL UPD(8),PUPD(8,8),QFDMAX(8),QFDMIN(8)
      COMPLEX SD(24,24),SX(24,24),SY(24,24)
      INTEGER NN(2)
      COMPLEX DATA(24,24),WORK(50),SAVE(24,24),DX(24,24),DY(24,24)
      COMPLEX SDATA(24,24)
C
C       DATA STRUCTURES TO GATHER STATISTICS ON FILTER
C                      TRACKER CAPIBILITY
C     XFME   IS THE ERROR BETWEEN THE PREDICTED X DYNAMIC LOCATION
C             AT A PARTICULAR MINUS TIME AND THE TRUTH MODEL TRUE
C             X DYNAMIC LOCATION
C     XFME2  IS THE SQUARE OF THE XFME
C
C     NOTE THAT XFME AND XFME2 ARE ARRAYS WHICH ARE DIMENSIONED TO
C               BE 2X150 THE FIRST ROW IN EACH IS USED FOR THE X
C               DIRECTION WHILE THE SECOND ROW IS FOR THE Y DIRECTIO
C
C     CNME   IS THE ERROR IN THE PREDICTED LOCATION OF THE CENTROID
C            A PARTICULAR MINUS TIME COMPARED TO THE TRUTH MODEL
C     CNME2  IS THE SQUARE OF CNME
C
C     NOTE AGAIN THE DIMENSION OF CNME AND CNME2C
C
      REAL XFME(6,150),XFME2(6,150),CNME(2,150),CNME2(2,150)
C
C     XFPE   IS THE ERROR BETWEEN THE UPDATED DYNAMIC LOCATION AT A
C            PARTICULAR PLUS TIME AND THE TRUTH MODEL TRUE DYNAMIC
C     XFPE2  IS THE SQUARE OF XFPE
C
C     NOTE THE DIMENSIONALITY OF XFPE,XFPE2 FOR THE SAME REASONS AS
C            ABOVE THE 150 ALLOWS  COMPUTATION OF STATISTICS PER
C            FRAME UP TO 150 FRAME
C
C     CNPE   IS THE CENTROID ERROR AT THE PLUS TIME
C     CNPE2C
C
      REAL XFPE(6,150),XFPE2(6,150),CNPE(2,150),CNPE2(2,150)
      REAL PDM(8,150),PDP(8,150)
C
C           FILTERS DATA STRUCTURES
C
C     PHIF  IS THE STATE TRANSITION MATRIX FOR THE KALMAN FILTER
C          -SEE SUBROUTINE FILTER
C     QFD   IS THE RESULT OF THE INTEGRAL TERM IN THE PROPAGATION
C          -OF THE COV MATRIX SEE SUBROUTINE PROPF
C     PFP   IS THE FILTERS COVARIANCE MATRIX PLUS- AFTER INCORPORATI
C          -OF A MEASUREMENT
C     PFM   IS THE FILTERS COVARIANCE MATRIX MINUS AFTER PROPAGATION
```

```
C             -BUT PRIOR TO MEASUREMENT INCORPORATION
C        XFP   IS THE FILTER STATE VECTOR PLUS
C        XFM   IS THE FILTER STATE VECTOR MINUS
C        PDIAG   IS THE MATRIX THAT WILL PASS DIAGONAL ELEMENTS OF PFP A
C                PFM TO STATFP AND STATFM RESPECTIVELY
C
         REAL PHIF(8,8),QFD(8,8),PFP(8,8),PFM(8,8),XFP(8),XFM(8),PDIAG(8
         REAL XFPOLD(8),HTRR(2)
C
C        Z     IS THE KALMAN FILTER MEASUREMENT VECTOR
C
         REAL Z(64)
C
         DATA NN/24,24/
C
C        INITIALIZE THE FILTERS DATA STRUCTURES
C
         DATA DT/.0333333/
         DATA TDF/1.5/
C
C*****************************************************************************
C**                          I N I T I A L I Z A T I O N
C*****************************************************************************
C
         CALL RANSET(12345)
         PRINT *, "    ******************"
         PRINT *, "        PROGRAM GAUSS  "
         PRINT *, "    ******************"
560      FORMAT(F6.2)
54321    READ(5,561) NZ
         IF (EOF(5).NE.0) GO TO 6421
         WRITE(6,3791) NZ
3791     FORMAT(1X,*NUMBER OF ZEROES TO PAD =  *,I3)
561      FORMAT(I3)
         NZM=25-NZ
3792     FORMAT(1X,*NUMBER OF FRAMES =  *,I3)
         READ(5,561) NFRMS
         WRITE(6,3792) NFRMS
4023     FORMAT(1X,*NUMBER OF SIMULATIONS =  *,I3)
         READ(5,561) NRUNS
         WRITE(6,4023) NRUNS
3793     FORMAT(1X,*ALPHA FOR SMOOTHING =  *,F6.2)
         READ(5,560) ALPHA
         WRITE(6,3793) ALPHA
3795     FORMAT(1X,*NUMBER OF HIGH FREQ COMPONENTS TO ZERO =  *,I3)
         READ(5,561) NFREQ
         WRITE(6,3795) NFREQ
         ISF=14-NFREQ
         IEF=12+NFREQ
3789     FORMAT(1X,*INPUT MEASUREMENT ERROR VARIANCE =  *,F6.2)
         READ(5,560) VARM
         WRITE(6,3789) VARM
562      FORMAT(F10.2)
C
         CALL INITF(TAF,VARDF,VARAF,VARYQ,VARDFO)
C
C        DEFINE TRUTH MODEL DYNAMICS
```

```
C
965   FORMAT(1X,*STD. DEV OF TRUTH MODEL ATMOSPHERIC JITTER*,F5.3)
      READ(5,6666) SIGAT
      WRITE(6,965) SIGAT
6666  FORMAT(F5.3)
C
C     READ IN INITIAL CONDITIONS FOR THE VEHICLE POSITION AND VELOCIT
C
      DO 600 I=1,3
      DO 600 J=1,3
      READ(5,562) IC(I,J)
      RF(1,J)=FLOAT(J)
      RF(2,J)=0.
600   CONTINUE
      WRITE(6,6789) ((IC(I,J),J=1,3),I=1,3)
6789  FORMAT(1X,*INERTIAL POSITION*,/,5X,*XO= *,F10.2,5X,*YO= *,F10.2
     #   ,*ZO= *,F10.2,//,* INERTIAL VELOCITY*,/,2X,*XDOTO= *,F10.2
     #  *YDOTO= *,F10.2,2X,*ZDOTO= *,F10.2,//,*INERTIAL ACCELERATIC
     #  /,*AXO= *,F10.2,5X,*AYO= *,F10.2,5X,*AZO= *,F10.2,/)
C
C     SET UP THE DESCRIPTION OF THE TARGET AND ITS TRAJECTORY
C
      CALL DESCRIB(IMAX,XMAX,YMAX,SIGVO,SIGPVO,DELV,DELPV,
     #  S,REVRT,TO,T1,NG,YT,ITARG)
C
C
C
C
C     INITIALIZE TRUTH MODEL MATRICIES
C
      CALL TRUTH(PHIT,BD,UT,QDROOT,H,SIGAT,DT)
C
C     INTIALIZE THE FILTERS PARAMETERS
C
C     INITIALIZE THE FILTERS MATRICES DEFINITION
C
      CALL FILTER(TDF,VARDF,TAF,VARAF,DT,PHIF,QFD,QFDMAX,QFDMIN)
      CALL PRINT(QFDMIN,1,8,6HQFDMIN)
      CALL PRINT(QFDMAX,1,8,6HQFDMAX)
      CALL PRINT(QFD,8,8,3HQFD)
C
C
C
C     INITIALIZE THE FILTER ERROR MATRICES TO ZERO
C
      DO 23 J=1,NFRMS
      DO 21 I=1,2
      CNME(I,J)=CNPE(I,J)=0.
21    CNME2(I,J)=CNPE2(I,J)=0.
      DO 22 I=1,6
      XFME(I,J)=XFPE(I,J)=0.
22    XFPE2(I,J)=XFME2(I,J)=0.
      DO 23 I=1,8
23    PDM(I,J)=PDP(I,J)=0.
C
C     SET UP IDEAL DATA FOR ERROR CALCULATION
C
```

```
C     CALL IDEAL(IMAX,S,XMAX,YMAX,24,NZ,X,Y,SD,SX,SY,RF,NS)
C
C USINGFIRST AND SECOND NEAREST NEIGHBOR DETERMINE THE CHOLESKY
C SQUAREROOT, R, OF THE MEASUREMENT COVARIANCE MATRIX, R
C
      CALL SPTN(VARM,R,8)
C
C     THIS LOOP MAKES SPATIALLY CORRELATED/UNCORRELATED NOISE
C       COMMENT THE NEXT FOUR LINES IF WANT SPATIAL CORRELATION
      PRINT *, "RFIL = 2.*R"
      DO 6428 I=1,64
      DO 6428 J=1,64
      RFIL(I,J)=2.*R(I,J)
C     IF (I.NE.J) RFIL(I,J)=0.
C     RINV(I,J)=0.
C     IF (I.EQ.J) RINV(I,J)=1./RFIL(I,J)
6428  CONTINUE
      PRINT *, "DIAGONAL RFIL"
C
C     COMPUTE RROOT -  THE CHOLESKY SQRT OF R
C
      CALL CHOLY(R,64,RROOT)

C
C     GET THE INVERSE OF THE SPATIALLY CORRELATED MEAS NOISE COV
C     MATRIX USED IN THE FILTER MODEL
C      - NEEDED FOR THE INVERSE COV METHOD
C
      CALL INVERT(RFIL,64,RINV)
C
C**************************************************************
C**              E N D   I N I T I A L I Z A T I O N        **
C**************************************************************
C
C**************************************************************
C**   B E G I N   M O N T E C A R L O   S I M U L A T I O N   *
C**************************************************************
C
C     MAKE NRUNS SIMULATIONS OF NFRMS EACH FOR MONTE-CARLO ANALYSIS
C
91    FORMAT (1X,*RUN NUMBER *,I3,/)
      DO 90 NS=1,NRUNS
      WRITE(6,91) NS
C
      XSHIFT=0.
      YSHIFT=0.
      HTRR(1)=HTRR(2)=0.
C
C     ZERO OUT IDEAL AND ACTUAL DATA DERIVATIVE ARRAYS
C      INITIALIZE SMOOTHED DATA ARRAY
C
      DO 7 I=1,24
      DO 7 J=1,24
      DX(I,J)=CMPLX(0.,0.)
      DY(I,J)=CMPLX(0.,0.)
 7    SDATA(I,J)=CMPLX(0.,0.)
C
```

```
C     INITIALIZE STATE VECTOR
C
      DO 71 I=1,8
71    XT(I,1)=0.
      XT(1,1)=3.0
      XT(2,1)=3.0
      YT(1,1)=3.0
      YT(2,1)=3.0
C
C     INITIAL CONDITIONS FOR THE FILTER
C
      DO 106 I=1,8
      UPD(I)=0.0
      DO 106 J=1,8
      PFM(I,J)=0.0
      PFP(I,J)=0.0
      PUPD(I,J)=0.0
      XFP(I)=0.0
106   XFM(I)=0.0
      PFP(1,1)=.2
      PFP(2,2)=.2
      PFP(3,3)=16.
      PFP(4,4)=16.
      PFP(5,5)=200.
      PFP(6,6)=200.
      PFP(7,7)=.2
      PFP(8,8)=.2
C
      TRXXT=20.
C
      RHORO=SQRT(IC(1,1)**2+IC(1,3)**2)
C
C     GET INITIAL CONDITIONS ON DYNAMIC STATES
C
      CALL COMPUTE(XI,YI,ZI,VX,VY,VZ,RANGE,UT,0,T1,IC,NG,NS)
C
C
      XFP(1)=3.0
      XFP(2)=3.0
      XFP(3)=UT(1)
      XFP(4)=UT(2)
      XFP(5)=UT(3)
      XFP(6)=UT(4)
C
      DO 30 I=1,6
      XFM(I)=XFP(I)
30    CONTINUE
C
C
C     DEFINE UPPER-LEFT CORNER OF FOV
C
      X=XFP(1)-4.
      Y=XFP(2)-4.
C
C
C     TRACK TARGET FOR NFRAME FRAMES (TIME SLICES)
C
```

```
C
      IF (NS.EQ.1) CALL PRINT(XFP,1,8,10HINIT STATE)
C
      DO 90 NR=1,NFRMS
C
C     LOCATE XMAX, YMAX, AND DETERMINE S AND RF (RANGE FLAG) MATRICIE
C     WHEN DYNAMIC IMAGE IS DESIRED
C
      IF (ITARG.NE.1) GO TO 39
      CALL LOCATE(XMAX,YMAX,SIGV,SIGPV,YT,UT,S,RANGE,XI,YI,ZI,
     #           VX,VY,VZ,NR,T0,T1,REVRT,DELV,DELPV,SIGVO,SIGPVO,NG,
     #           EVO,EPVO,EV,EPV,RF,NS)
C
C
39    CONTINUE
C
C
C     GET MEASUREMENT NOISE ARRAY
C
      CALL NOISE(W,64)
      CALL MULT(RROOT,W,64,64,1,V)
C
C     GET MEASUREMENT DATA
C
C
C     IF (ITARG.EQ.2)
C    #CALL SINGLE(IMAX,S,YT(1,1),YT(2,1),XFM(1),XFM(2),DATA,DX,DY,NR,(
C     IF (ITARG.NE.2)
      CALL INPUT3(IMAX,S,XMAX,YMAX,24,NZ,X,Y,DATA,CENX,CENY,RF,NR,NS)
C
C     CALL IDEAL(IMAX,S,XMAX,YMAX,24,NZ,X,Y,DATA,DX,DY,RF,ITARG,NR,NS
C     ADD CORRELATED MEASUREMENT NOISE TO CENTER 8X8 PIXEL DATA
C
      DO 4 I=1,8
      DO 4 J=1,8
      DATA(I+8,J+8)=DATA(I+8,J+8)+CMPLX(V(8*(I-1)+J),0.0)
4     CONTINUE
C
C     ADD UNCORRELATED NOISE TO MEASUREMENT DATA OUTSIDE CENTER
C     8X8 PIXEL AREA.
C
      CALL NOISE(UC,576)
      DO 6 I=1,24
      DO 6 J=1,24
      IF(I.GE.9.AND.I.LE.16.AND.J.GE.9.AND.J.LE.16) GO TO 6
      IF((I.LE.NZ).OR.(J.LE.NZ).OR.(I.GE.NZM).OR.(J.GE.NZM)) GO TO 6
      DATA(I,J)=DATA(I,J)+CMPLX(UC(24*(I-1)+J),0.)*SQRT(VARM)
6     CONTINUE
C
C
C     CREATE THE MEASUREMENT VECTOR FOR THE FILTER UPDATE
C
      K=0
      DO 101 I=9,16
      DO 101 J=9,16
      K=K+1
      Z(K)=REAL(DATA(I,J))
```

```
C
C     LOWER BOUND MEASUREMENT
C
      IF (Z(K).LT.0.) Z(K)=0.
      Z(K)=Z(K)+0.1
101   CONTINUE
C
C     SAVE DIAGONAL ELEMENTS OF PFM FOR FILTER SIGMA PLOTS
      DO 111 I=1,8
111   PDIAG(I)=PFM(I,I)
C
C     GO CALCULATE THE ERRORS OF THE FILTERS ESTIMATE PRIOR TO
C        MEASUREMENT INCORPORATION
      CALL STATFM(XFME,XFME2,CNME,CNME2,XFM,XT,YT,UT,NR,NFRMS,
     #    PDIAG,PDM)
C
C
      IF(NP.EQ.1) GO TO 164
      MANIND=0
C
C     SHIFT TO SAVE COPY OF OLD XFP
C
      DO 400 I=1,8
400   XFPOLD(I)=XFP(I)
C
C     INCORPORATE MEASUREMENT
C
200   CALL UPDAT(Z,LINH,NLINH,XFP,XFM,PFP,PFM,UPD,RINV,NS,NR,HTRR)
C     DETERMINE TRXXT AND APPROPRIATE QFD
C
      TRXXTO=TRXXT
      CALL QADAPT(NR,TRXXT,UPD,PUPD,QFD,QFDMIN,QFDMAX,PFP,
     #     VARDFO,TDF,VARAF,TAF,VARYQ,NS)
C
C
      GO TO 300
C
C
C     FOLLOWING STATEMENT PREVENTS RE-PROCESSING ESTIMATE TWICE
C
      IF (MANIND.EQ.1) GO TO 300
      IF (TRXXT.LT.5.*TRXXTO) GO TO 300
C
C     SET MANEUVER INDICATOR
C
      MANIND=1
      PRINT *, "ESTIMATE REPROCESSED"
C
C     LOWER BOUND PFP DURING MANEUVER
C
C     THIS PROCESS MAINTAINS THE SAME CORRELATION COEFFICIENTS
C
      DO 233 I=3,4
      VFACTOR=10./SQRT(PFP(I,I))
      AFACTOR=300./SQRT(PFP(I+2,I+2))
      DO 232 J=1,8
      IF (PFP(I,I).LT.100.)PFP(I,J)=PFP(I,J)*VFACTOR
      IF(PFP(I,I).LT.100.) PFP(I,J)=PFP(I,J)*VFACTOR
      IF (PFP(I+2,I+2).LT.90000.) PFP(I+2,J)=PFP(I+2,J)*AFACTOR
      IF (PFP(I+2,I+2).LT.90000.) PFP(J,I+2)=PFP(J,I+2)*AFACTOR
232   CONTINUE
233    CONTINUE
C
C     REPROPOGATE OLD ESTIMATE USING INCREASED PFP
C
      CALL PROPF((NR-1),NS,PHIF,QFD,PFP,PFM,XFPOLD,XFM,PUPD,MANIND)
C
C     UPDATE RE-PROPOGATED ESTIMATE
C
      GO TO 200
300   MANIND=0
      DO 222 I=1,8
222   PDIAG(I)=PFP(I,I)
C
C     CALCULATE THE ERRORS FOR THE FILTER AFTER THE INCORPORATION
C            OF THE MEASUREMENT
      CALL STATFP(XFPE,XFPE2,CNPE,CNPE2,XT,YT,UT,XFP,NR,NFRMS,
     #    PDIAG,PDP)
C
C     COMPUTE THE SHIFT INFORMATION FROM THE CENTER OF FOV
      XSHIFT=X-XFP(1)+4.-XFP(7)
      YSHIFT=Y-XFP(2)+4.-XFP(8)
C
      IF (NS.EQ.1) WRITE(6,175) XSHIFT,YSHIFT
175   FORMAT(T12,*XSHIFT* :*,F10.7,T42,*YSHIFT* :*,F10.7)
C
164   CONTINUE
C
C     SHIFT THE DATA ARRAY APPROPRIATELY
C
C     GET FORWARD FFT
C
      CALL FOURT(DATA,NN,2,-1,1,WORK)
C
      IF (NS.EQ.1)
     #WRITE(6,165)NR,YT(1,1),YT(2,1),(UT(I),I=1,4),(XFM(I),I=1,6),
     #              (XFP(I),I=1,6)
165   FORMAT(/,T2,*FRAME: *,I3,8X,*X POS*,10X,*Y POS*,10X,*X VEL*,10X
     #      *Y VEL*,9X,*X ACCEL*,8X,*Y ACCEL*,
     #      /,*  TRUTH MODEL:*,6(1X,F14.5),
     #      /,* FILTER MINUS:*,6(1X,F14.5),
     #      /,*  FILTER PLUS:*,6(1X,F14.5))
      IF (NS.EQ.1) PRINT *,"        TRXXT= ",TRXXT,"  CENX= ",
     #CENX,"  CENY= ",CENY
C
C
C     FILTER DESIRED FREQUENCY COMPONENTS OUT
C
      IF(NFREQ.GT.12) NFREQ=12
      IF(NFREQ.LE.0) GO TO 3796
      DO 8 I=ISF,IEF
      DO 8 J=1,24
      DATA(I,J)=CMPLX(0.,0.)
 8    DATA(J,I)=CMPLX(0.,0.)
 3796 CONTINUE
C
C     ASSUME IF NR=1 THAT THE DATA IS CENTERED
      IF(NR.NE.1) CALL SHIFT(DATA,24,XSHIFT,YSHIFT)
      CALL SMOOTH(DATA,SDATA,ALPHA,24,NR)
C
C     IF (NR.EQ.10.AND.NS.EQ.1) CALL DISPLAY(24,24,SX)
C
C     IF (ITARG.EQ.1.AND.NR.EQ.((NR/5)*5).AND.NS.EQ.1)
C    #  CALL DISPLAY(24,24,SDATA)
      IF (NR.EQ.10.AND.NS.EQ.1) CALL DISPLAY(24,24,DX)
C     IF (NR.EQ.10.AND.NS.EQ.1) CALL DISPLAY(24,24,SY)
      IF (NR.EQ.10.AND.NS.EQ.1) CALL DISPLAY(24,24,DY)
C
      CALL PROPF(NR,NS,PHIF,QFD,PFP,PFM,XFP,XFM,PUPD,MANIND)
C
C
C
      X=XFM(1)-4.
      Y=XFM(2)-4.
      XSHIFT=XFM(7)
      YSHIFT=XFM(8)
C
C     ROUTINE FOR COMPUTING PERFECT INTENSITY DATA FOR ERROR COMPUTATI
C
      DO 170 I=1,24
      DO 170 J=1,24
  170 SAVE(I,J)=SDATA(I,J)
      CALL SHIFT(SAVE,24,XSHIFT,YSHIFT)
      CALL DERIV(SAVE,24,DX,DY)
      CALL FOURT(SAVE,NN,2,1,1,WORK)
      CALL FOURT(DX,NN,2,1,1,WORK)
      CALL FOURT(DY,NN,2,1,1,WORK)
C
C     SCALE THE INVERSE TRANSFORM ALONG WITH SETTING THE APPROPRIATE
C     SIGN TO INDICATE THE CHANGE IN INTENSITY PATTERN WITH PESPECT TO
C     CHANGE IN STATE
C
      DO 172 I=1,24
      DO 172 J=1,24
      DX(I,J)=-DX(I,J)/576.
      SAVE(I,J)=SAVE(I,J)/576.
  172 DY(I,J)=-DY(I,J)/576.
C
C     FILL IN THE LINEARIZED INTENSITY ARRAY
C
C     IF ITARG=3 GIVE PERFECT KNOWLEDGE OF DX AND DY
C     IF (ITARG.EQ.2)
C    # CALL SINGLE(IMAX,S,XSHIFT,YSHIFT,0.,0.,DATA,DX,DY,NR,NS)
C
C
      K=0
      DO 102 I=9,16
      DO 102 J=9,16
      K=K+1
      LINH(K,1)=REAL(DX(I,J))
      LINH(K,2)=REAL(DY(I,J))
      LINH(K,3)=0.
      LINH(K,4)=0.
      LINH(K,5)=0.
      LINH(K,6)=0.
      LINH(K,7)=LINH(K,1)
      LINH(K,8)=LINH(K,2)
102   NLINH(K)=REAL(SAVE(I,J))
C
      IF (NR.NE.((NR/5)*5).AND.(NR.LT.60.OR.NR.GT.80)) GO TO 105
      IF (NS.EQ.1 ) WRITE(6,5679) ((REAL(DX(I,J)),
     # J=9,16),I=9,16)
5679  FORMAT(* DX *,8(T10,8F6.2,/))
C
C
      IF (NS.EQ.1 ) WRITE(6,5680) ((REAL(DY(I,J)),
     # J=9,16),I=9,16)
5680  FORMAT(* DY *,8(T10,8F6.2,/))
C
      IF (NS.EQ.1 ) WRITE(6,5677) Z, NLINH
     #,((Z(I)-NLINH(I)),I=1,64)
5677  FORMAT(* Z*,8(T10,8F6.2,/),/,* NLINH*,8(T10,8F6.2,/),
     #/,* Z-NLINH*,8(T10,8F6.2,/))
C
C
105   CALL PROP(PHIT,BD,UT,QDROOT,H,XT,YT,8,2,NR,DT,IC,NG,NS,
     #            RANGE,XI,YI,ZI,VX,VY,VZ,T1)
C
C
C     PRINT LINE INDCATING END OF PRINTED INFO FOR THIS FRAME
      IF(NS.EQ.1)PRINT *, "--------------------------------------------
     #-----------------------------------------------------------------
      IF(NS.EQ.1)PRINT *, " "
C
      IF (ITARG.EQ.1) GO TO 90
C
C     COMPUTE NEW XMAX   YMAX POSITIONS FOR STATIC IMAGE TARGET AND
C     DEFINE GAUSSIAN PEAK LOCATIONS BASED ON CENTROID POSITION, YT
C
C
      IF (ITARG.NE.0) GO TO 89
      XMAX(1)=YT(1,1)
      XMAX(2)=YT(1,1)-2.
      XMAX(3)=YT(1,1)+2.
      YMAX(1)=YT(2,1)-2.666666
      YMAX(2)=YT(2,1)+1.333333
      YMAX(3)=YT(2,1)+1.333333
      GO TO 90
89    XMAX(1)=XMAX(2)=XMAX(3)=YT(1,1)
      YMAX(1)=YMAX(2)=YMAX(3)=YT(2,1)
C
C
C
90    CONTINUE
C*********************************************************************
C**   E N D   M O N T E C A R L O   S I M U L A T I O N           **
C*********************************************************************
C
```

```
            C      WRITE SOME OF THE PLOT G9INF TO TAPE8
            1000   FORMAT(5I3)
            2000   FORMAT(6E12.5)
                   WRITE(8,1000) NG,NFRMS,NRUNS,ITARG
 575               WRITE(8,2000) ALPHA,SIGAT,VARM,VARDFO
            C
            C      CALCULATE MEAN AND VARIANCE STATISTICS
            C
                   CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,
 580              #NFRMS,PDM,PDP)
            C
                   WRITE(6,9987) NRUNS,NFRMS,NZ,NFREQ,COV,VARM,ALPHA,
                  #                      SIGAT
            9987   FORMAT(1H1,T10,*RUNS=*,I3,T38,*FRAMES=*,I3,T73,*NUMBER ZERO PAD
 585              #    I1,T100,*NUMBER FREQ ZEROED=*,I3,/,T10,*GAUSSIAN COVARIANCE
                  #F5.2,T38,*MEASUREMENT NOISE VARIANCE=*,F5.1,T73,*SMOOTHING ALPH
                  # =*,F7.3,/,T10,
                  # *TRUTH MODEL UNCERTAINTY=*,F7.3,///)
            6421   STOP
 590               END
```

```
NR. SEVERITY  DETAILS    DIAGNOSIS OF PROBLEM

 583    I                   THERE IS NO PATH TO THIS STATEMENT.
```

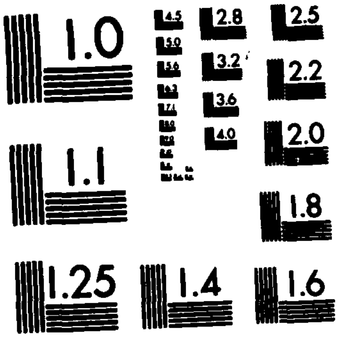
```
      SUBROUTINE DESCRIB(IMAX,XMAX,YMAX,SIGVO,SIGPVO,DELV,DELPV,S,REV
     #         TO,T1,NG,YT,ITARG)
      REAL IMAX(3),XMAX(3),YMAX(3),SIGVO(3),SIGPVO(3),DELV(3),DELPV(3
      REAL S(12),YT(2,1)
C
C
C     THIS SUBROUTINE CONTROLS SOME OF THE INPUT VALUES NEEDED TO
C     DESCRIBE THE TARGET TO BE TRACKED AND THE MANEUVER TO BE PERFOR
C
C     INITIALIZE TARGET INTENSITY ASSUMING
C     3 CIRCULAR CROSS-SECTION GAUSSIAN TARGET.
C
      IMAX(1)=20.
      IMAX(2)=20.
      IMAX(3)=20.
      READ(5,561) NG
      WRITE(6,6000) NG
6000  FORMAT(1X,*NUMBER OF G TURN =*,I3,/)
C
      READ(5,563) TO,T1,REVRT
      WRITE(6,6050) TO,REVRT,T1
6050  FORMAT(1X,*STARTING TIME OF ROLL MANEUVER =*,F7.3,/,1X,
     #       *ROLL RATE (REV/SEC) =*,F7.3,/,1X,*STARTING TIME OF
     #G-PULLING MANEUVER =*,F7.3)
C
562   FORMAT(F10.2)
561   FORMAT(I3)
563   FORMAT(3F7.3)
C
C     READ INDICATOR THAT TELLS WHETHER TARGET IMAGE IS TO BE STATIC
C     OR DYNAMIC...1: DYNAMIC, 2 OR 0: STATIC.
C     ITARG=2  IMPLIES SINGLE SPOT TARGET, ITARG= 0 OR 1  IMPLIES
C     THREE SPOT TARGET
C
      READ(5,561) ITARG
      IF (ITARG.EQ.1) PRINT *, "DYNAMIC TARGET IMAGE."
      IF (ITARG.NE.1) PRINT *, "STATIC TARGET IMAGE."
      IF (ITARG.NE.1) GO TO 100
C
      READ (5,563) SIGVO
      WRITE(6,6100) SIGVO
6100  FORMAT(1X,*GAUSSIAN DISPERSION VALUES IN VEL DIRECTION = *,3F7.
      READ(5,563) SIGPVO
      WRITE(6,6200) SIGPVO
6200  FORMAT(1X,*GAUSSIAN DISPERSION VALUES PERP TO VEL. = *,3F7.3)
      READ(5,563) DELV
      WRITE(6,6300) DELV
6300  FORMAT(1X,*DISTANCES OF ELLIPSOID CENTERS FROM CG OF TARGET IN
     #VEL. DIR. = *,3F7.3)
      READ(5,563) DELPV
      WRITE(6,6400) DELPV
6400  FORMAT(1X,*DISTANCES OF ELLIPSOID CENTERS FROM CG IN DIR. PERP
     #  TO THE VEL = *,3F7.3)
      IF (ITARG.EQ.1) GO TO 999
100   CONTINUE
C
C
```

```
C   DEFINE TRUE TARGET AS 3 INDEPENDENT GAUSSIAN FUNCTIONS WITH
C     DISPERSION PARAMETER = COV WHEN STATIC IMAGE DESIRED, OTHERWISE
C     WILL BE SET UP EACH FRAME WITH A CALL TO SUBROUTINE LOCATE.
C
      COV=3.0
      S(1)=1./COV
      S(2)=0.
      S(3)=0.
      S(4)=S(1)
      S(5)=S(1)
      S(6)=0.
      S(7)=0.
      S(8)=S(1)
      S(9)=S(1)
      S(10)=0.
      S(11)=0.
      S(12)=S(1)
C
C     DEFINE GAUSSIAN PEAK LOCATIONS BASED ON CENTROID POSITION, YT
C     FOR STATIC TARGET IMAGE
C
      YT(1,1)=YT(2,1)=3.0
C
      IF (ITARG.NE.0) GO TO 200
      XMAX(1)=YT(1,1)
      XMAX(2)=YT(1,1)-2.
      XMAX(3)=YT(1,1)+2.
      YMAX(1)=YT(2,1)-2.666666
      YMAX(2)=YT(2,1)+1.333333
      YMAX(3)=YT(2,1)+1.333333
C
      GO TO 999
200   XMAX(1)=XMAX(2)=XMAX(3)=YT(1,1)
      YMAX(1)=YMAX(2)=YMAX(3)=YT(2,1)
C
999   RETURN
      END
      SUBROUTINE STATFM(XFME,XFME2,CNME,CNME2,XFM,XT,YT,UT,NR,NFRMS,
     #    PDIAG,PDM)
      REAL XFME(6,NFRMS),XFME2(6,NFRMS),CNME(2,NFRMS),CNME2(2,NFRMS)
      REAL XFM(8),XT(8,1),YT(2,1)
      REAL UT(4),PDIAG(8),PDM(8,NFRMS)
C
C     THIS ROUTINE GATHERS THE INFORMATION THAT WILL BE
C        REQUIRED TO COMPUTE THE STATISTICS OF THE PREDICTIONS
C        OF THE FILTER PRIOR TO MEASUREMENT INCORPORATION
C
C     FIRST COLLECT THE ERROR IN THE PREDICTED DYNAMIC LOCATION
C
      DIF3=XFM(3)-UT(1)
      DIF4=XFM(4)-UT(2)
      DIF5=XFM(5)-UT(3)
      DIF6=XFM(6)-UT(4)
C
      XFME(1,NR)=XFME(1,NR)+XFM(1)-XT(1,1)
      XFME(2,NR)=XFME(2,NR)+XFM(2)-XT(2,1)
      XFME(3,NR)=XFME(3,NR)+DIF3
      XFME(4,NR)=XFME(4,NR)+DIF4
      XFME(5,NR)=XFME(5,NR)+DIF5
      XFME(6,NR)=XFME(6,NR)+DIF6
C
C     NOW COLLECT THE SQUARE OF THAT ERROR
C
      XFME2(1,NR)=XFME2(1,NR)+(XFM(1)-XT(1,1))**2
      XFME2(2,NR)=XFME2(2,NR)+(XFM(2)-XT(2,1))**2
      XFME2(3,NR)=XFME2(3,NR)+DIF3**2
      XFME2(4,NR)=XFME2(4,NR)+DIF4**2
      XFME2(5,NR)=XFME2(5,NR)+DIF5**2
      XFME2(6,NR)=XFME2(6,NR)+DIF6**2
C
C     COLLECT ERROR IN CENTROID PREDICTED LOCATION MINUS
C
      CNME(1,NR)=CNME(1,NR)+(XFM(1)+XFM(7)-YT(1,1))
      CNME(2,NR)=CNME(2,NR)+(XFM(2)+XFM(8)-YT(2,1))
C
C     COLLECT THE SQUARE OF THE ERROR
C
      CNME2(1,NR)=CNME2(1,NR)+(XFM(1)+XFM(7)-YT(1,1))**2
      CNME2(2,NR)=CNME2(2,NR)+(XFM(2)+XFM(8)-YT(2,1))**2
      DO 100 I=1,8
      PDM(I,NR)=PDM(I,NR)+SQRT(PDIAG(I))
100   CONTINUE
C
C     ADD DYNAMICS TERM TO THAT REPRESENTING CENTROID FILTER SIGMA
C
      DO 200 I=1,2
200   PDM(I+6,NR)=PDM(I+6,NR)+SQRT(PDIAG(I))
C
      RETURN
      END
      SUBROUTINE STATFP(XFPE,XFPE2,CNPE,CNPE2,XT,YT,UT,XFP,NR,NFRMS,
     #    PDIAG,PDF)
      REAL XFPE(6,NFRMS),XFPE2(6,NFRMS),CNPE(2,NFRMS),CNPE2(2,NFRMS)
      REAL XT(8,1),YT(2,1),XFP(8)
      REAL UT(4),PDP(8,NFRMS),PDIAG(8)
C
C     THIS ROUTINE GATHERS THE INFORMATION THAT WILL BE
C        REQUIRED TO COMPUTE THE STATISTICS ON THE FILTERS
C        UPDATED STATE ESTIMATES
C
C     COMPUTE DIFFERENCES THAT WILL BE NEEDED
C
      DIF1=XFP(1)-XT(1,1)
      DIF2=XFP(2)-XT(2,1)
      DIFF3=XFP(1)+XFP(7)-YT(1,1)
      DIFF4=XFP(2)+XFP(8)-YT(2,1)
      DIF3=XFP(3)-UT(1)
      DIF4=XFP(4)-UT(2)
      DIF5=XFP(5)-UT(3)
      DIF6=XFP(6)-UT(4)
C
C
C     FIRST COLLECT THE ERROR IN THE DYNAMIC LOCATION ESTIMATES
C
      XFPE(1,NR)=XFPE(1,NR)+DIF1
      XFPE(2,NR)=XFPE(2,NR)+DIF2
      XFPE(3,NR)=XFPE(3,NR)+DIF3
      XFPE(4,NR)=XFPE(4,NR)+DIF4
      XFPE(5,NR)=XFPE(5,NR)+DIF5
      XFPE(6,NR)=XFPE(6,NR)+DIF6
C
C     NOW COLLECT THE SQUARE OF THAT ERROR
C
      XFPE2(1,NR)=XFPE2(1,NR)+DIF1**2
      XFPE2(2,NR)=XFPE2(2,NR)+DIF2**2
      XFPE2(3,NR)=XFPE2(3,NR)+DIF3**2
      XFPE2(4,NR)=XFPE2(4,NR)+DIF4**2
      XFPE2(5,NR)=XFPE2(5,NR)+DIF5**2
      XFPE2(6,NR)=XFPE2(6,NR)+DIF6**2
C
C     NOW COLLECT THE ERROR IN THE CENTROID UPDATE
C
      CNPE(1,NR)=CNPE(1,NR)+DIFF3
      CNPE(2,NR)=CNPE(2,NR)+DIFF4
C
C     NOW COLLECT THE ERROR SQUARED
C
      CNPE2(1,NR)=CNPE2(1,NR)+DIFF3**2
      CNPE2(2,NR)=CNPE2(2,NR)+DIFF4**2
      DO 100 I=1,8
      PDP(I,NR)=PDP(I,NR)+SQRT(PDIAG(I))
100   CONTINUE
C
C     ADD DYNAMICS AND ATM FILTER SIGMAS TO GET CENTROID SIGMA
C
      DO 200 I=1,2
200   PDP(I+6,NR)=PDP(I+6,NR)+SQRT(PDIAG(I))
```

```
      C
              RETURN
50            END
```

AD-A124 781 ENHANCED IMAGE TRACKING: ANALYSIS OF TWO ACCELERATION 6/6
MODELS IN TRACKING..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. M R KOZEMCHAK
UNCLASSIFIED DEC 82 AFIT/GEO/EE/82D-4 F/G 17/7 NL

END
FILMED
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```
      SUBROUTINE FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,
     #NRUNS,NFRMS,PDM,PDP)
      REAL XFME(6,NFRMS),XFME2(6,NFRMS),XFPE(6,NFRMS),XFPE2(6,NFRMS)
      REAL CNME(2,NFRMS),CNME2(2,NFRMS),CNPE(2,NFRMS),CNPE2(2,NFRMS)
      REAL PDM(8,NFRMS),PDP(8,NFRMS)
C
C     EXPLANATION OF DATA STRUCTURES
C     XFME IS THE XPOS OF THE FILTER AT MINUS TIME ERROR
C     THESE ARE ALL COMPATIBLE WITH THE ABOVE ROUTINES
C     ALL NAMES FOLLOW THIS CODE
C
C
C     THIS ROUTINE COMPUTES THE STATISTICS ON THE FILTER ERRORS
C
      IF (NRUNS.EQ.1) GO TO 10
      A=FLOAT(NRUNS)
      DO 1 J=1,NFRMS
      DO 205 I=1,6
      XFME(I,J)=XFME(I,J)/A
      XFPE(I,J)=XFPE(I,J)/A
      XFME2(I,J)=SQRT(ABS(XFME2(I,J)-A*XFME(I,J)**2)/(A-1.))
205   XFPE2(I,J)=SQRT(ABS(XFPE2(I,J)-A*XFPE(I,J)**2)/(A-1.))
      DO 210 I=1,2
      CNME(I,J)=CNME(I,J)/A
      CNPE(I,J)=CNPE(I,J)/A
      CNME2(I,J)=SQRT(ABS(CNME2(I,J)-A*CNME(I,J)**2)/(A-1.))
210   CNPE2(I,J)=SQRT(ABS(CNPE2(I,J)-A*CNPE(I,J)**2)/(A-1.))
      DO 1 I=1,8
      PDM(I,J)=PDM(I,J)/A
1     PDP(I,J)=PDP(I,J)/A
C
C     NOTE THE ORDER OF OUTPUT TO TAPE8
C
C
      WRITE(8,99) (((XFME(I,J),XFME2(I,J),PDM(I,J),XFPE(I,J),XFPE2(I,
     #             PDP(I,J)),J=1,NFRMS),I=1,6)
      WRITE(8,99) (((CNME(I,J),CNME2(I,J),PDM(I+6,J),CNPE(I,J),
     #             CNPE2(I,J),PDP(I+6,J)),J=1,NFRMS),I=1,2)
10    CONTINUE
C
      PRINT *, "FILTER MEAN ERRORS PRIOR TO MEASUREMENT UPDATE:"
      WRITE(6,2) ((J,(XFME(I,J),I=1,6)),J=1,NFRMS)
2     FORMAT(*1FRAME*,3X,*X POSITION*,5X,*Y POSITION*,5X,*X VELOCITY*
     #     5X,*Y VELOCITY*,6X,*X ACCEL.*,8X,*Y ACCEL.*,/,
     #     200(/,I6,6(1X,F14.6)),/)
C
      PRINT *, "FILTER MEAN ERRORS AFTER MEASUREMENT UPDATE:"
      WRITE(6,2) ((J,(XFPE(I,J),I=1,6)),J=1,NFRMS)
99    FORMAT(3F15.6,10X,3F15.6)
C
      RETURN
      END
      SUBROUTINE TRUTH(PHIT,BD,UT,QDROOT,H,SIGAT,DT)
      REAL PHIT(8,8),BD(8,4),QD(8,8),K,QDROOT(8,8),H(2,8)
      REAL QDP(6,6),QDPRT(6,6)
      REAL UT(4)
C
C
C                : XT  :
C                : YT  :
C                : X1A :
C           XD=  : X2A :
C                : X3A :
C                : Y1A :
C                : Y2A :
C                : Y3A :
C
C
C     THE SOLUTION TO THE TRUTH MODEL STATE SPACE EQUATIONS IS
C
C
C        XD(I+1)= PHIT*XD(I) + BD*UT + SQRT(QD)*WT
C
C         WHERE PHIT=STATE TRANSITION MATRIX
C               SIGAT= ATMOS NOISE STANDARD DEVIATION
C               UT= THE CONTROL INPUT
C               BD= THE INPUT MATRIX FOR THE CONTROL
C               WT= GAUSSIAN NOISE VECTOR
C               QD= COVARIANCE MATRIX
C
C
      K=.382109544*SIGAT
      TD=1.
      A=14.14
      B=659.5
C
C     ZERO ALL MATRICES
C
      DO 1 I=1,8
      DO 1 J=1,8
      PHIT(I,J)=0.
      QDROOT(I,J)=0.
      QD(I,J)=0.
      IF (I.GT.2) GO TO 5
      H(I,J)=0.
    5 IF (J.GT.4) GO TO 1
      BD(I,J)=0.
    1 CONTINUE
C
C
      PHIT(1,1)=1.
      PHIT(2,2)=PHIT(1,1)
      PHIT(3,3)=EXP(-A*DT)
      PHIT(4,4)=EXP(-B*DT)
      PHIT(4,5)=DT*EXP(-B*DT)
      PHIT(5,5)=EXP(-B*DT)
      PHIT(6,6)=EXP(-A*DT)
      PHIT(7,7)=EXP(-B*DT)
      PHIT(7,8)=DT*EXP(-B*DT)
      PHIT(8,6)=EXP(-B*DT)
C
C
      BD(1,1)=DT
      BD(2,2)=DT
      BD(1,3)=DT*DT*.5
      BD(2,4)=DT*DT*.5
C
      UT(1)=UT(2)=UT(3)=UT(4)=0.0
C
      FACT=(K**2)*(A**2)*(B**4)
      FACT1=A-B
      FACT2=A+B
      FACT3=2.*B
      GS1=FACT/(FACT1**4)
      GS2=FACT/(FACT1**3)
      GS3=FACT/(FACT1**2)
      P1=1.-EXP(-2.*A*DT)
      P2=1.-EXP(-FACT2*DT)
      P3=1.-EXP(-2.*B*DT)
      P4=DT*EXP(-FACT2*DT)
      P5=DT*EXP(-2.*B*DT)
C
C
      QD(1,1)=0.
      QD(2,2)=QD(1,1)
      QD(3,3)=(GS1*P1)/(2.*A)
      QD(3,4)=P2*(GS2/FACT2**2-GS1/FACT2)-P4*GS2/FACT2
      QD(3,5)=GS2*P2/FACT2
      QD(4,3)=QD(3,4)
      QD(4,4)=P3*(GS1/FACT3-2.*GS2/FACT3**2+2.*GS3/FACT3**3)-
     #          P5*(-GS2/B+GS3*DT/FACT3+2.*GS3/FACT3**2)
      QD(4,5)=P3*(GS3/FACT3**2-GS2/FACT3)-P5*GS3/FACT3
      QD(5,3)=QD(3,5)
      QD(5,4)=QD(4,5)
      QD(5,5)=P3*GS3/FACT3
C
C    FILL OUT REST OF QD MATRIX
      DO 2 I=3,5
      DO 2 J=3,5
      QD(I+3,J+3)=QD(I,J)
2     CONTINUE
C
C     TAKE CHOLESKY SQRT OF NON-DIAGONAL PORTION OF QD MATRIX
C
      DO 864 I=1,6
      DO 864 J=1,6
864   QDP(I,J)=QD(I+2,J+2)
      CALL CHOLY(QDP,6,QDPRT)
      DO 865 I=1,6
      DO 865 J=1,6
865   QDROOT(I+2,J+2)=QDPRT(I,J)
C
      H(1,1)=1.
      H(1,3)=1.
      H(1,4)=1.
      H(2,2)=1.
```

```
15             H(2,6)=1.
               H(2,7)=1.
         C
               RETURN
               END
```

```
      SUBROUTINE PROP(PHIT,BD,UT,QDROOT,H,XT,YT,N,M,FRAME,DT,IC,NG,NS
     #          RANGE,XI,YI,ZI,VX,VY,VZ,T1)
      REAL PHIT(N,N),BD(N,4),UT(4),QDROOT(N,N),XT(N,1),YT(M,1),H(M,N
      REAL TEMP1(8,1),TEMP2(8,1),TEMP3(8,1)
      REAL IC(3,3)
      INTEGER FRAME
C
12367 FORMAT(//,(2X,8E10.3))
C
C     THIS ROUTINE IMPLEMENTS THE STATE TRANSITION EQUATION.
C
C         XT(I+1)=PHIT*XT(I) + BD*UT(I) + QDROOT*WD
C
C     WHERE    XT= STATE VECTOR (NX1)
C             PHIT= STATE TRANSITION MATRIX (NXN)
C             BD=DETERMINISTIC INPUT MATRIX
C             QDROOT= STATE UNCERTAINTY COVARIANCE MATRIX (NXN)
C             WD= GAUSSIAN DISTRIBUTED NOISE VECTOR (NX1)
C
C     AND  THE OUTPUT EQUATION
C
C             YT=H*XT
C
C     WHERE    YT=MEASUREABLE OUTPUT VECTOR (MX1)
C            H= STATE TO OUTPUT MAXTRIX (MXN)
C
      CALL COMPUTE(XI,YI,ZI,VX,VY,VZ,RANGE,UT,FRAME,T1,IC,NG,NS)
C
      CALL NOISE(TEMP1,N)
      CALL MULT(QDROOT,TEMP1,N,N,1,TEMP2)
      CALL MULT(PHIT,XT,N,N,1,TEMP1)
      CALL MULT(BD,UT,N,4,1,TEMP3)
      DO 1 I=1,N
1     XT(I,1)=TEMP1(I,1)+TEMP2(I,1)+TEMP3(I,1)
      CALL MULT(H,XT,M,N,1,YT)
      RETURN
      END
      SUBROUTINE COMPUTE(X,Y,Z,VX,VY,VZ,RANGE,UT,FRAME,T1,IC,NG,NS)
      REAL UT(4),IC(3,3)
      INTEGER FRAME
C
C     THIS ROUTINE COMPUTES THE APPROPRIATE DETERMINISTIC
C     CONTROL INPUTS
C
      DT=1./30.
      X0=IC(1,1)
      Y0=IC(1,2)
      Z0=IC(1,3)
      VX0=IC(2,1)
      VY0=IC(2,2)
      VZ0=IC(2,3)
      AX0=IC(3,1)
      AY0=IC(3,2)
      AZ0=IC(3,3)
C
      V0=SQRT(VX0**2+VY0**2+VZ0**2)
      OMEGA=9.8*NG/V0
      VHOR0=SQRT(VX0**2+VZ0**2)
      T=FLOAT(FRAME)*DT
C     FIND WHAT PART OF TRAJECTORY TARGET LOCATED ON AND
C     DETERMINE APPROPRIATE ACCELERATION, VELOCITY AND POSITION.
C
      IF (T.GE.T1) GO TO 10
C
      X=X0+VX0*T+.5*T*T*AX0
      Y=Y0+VY0*T+.5*T*T*AY0
      Z=Z0+VZ0*T+.5*T*T*AZ0
C
      VX=T*AX0+VX0
      VY=T*AY0+VY0
      VZ=T*AZ0+VZ0
C
      AX=AX0
      AY=AY0
      AZ=AZ0
      GO TO 50
C
   10 ARG1=OMEGA*(T-T1)
C
      X=X0+T1*VX0+.5*T1*T1*AX0+((VX0*V0/VHOR0)/OMEGA)*SIN(ARG1)
      Y=Y0+T1*VY0+.5*T1*T1*AY0+(V0/OMEGA)*(1.-COS(ARG1))
      Z=Z0+T1*VZ0+.5*T1*T1*AZ0+((VZ0*V0/VHOR0)/OMEGA)*SIN(ARG1)
C
      VX=(VX0*V0/VHOR0)*COS(ARG1)
      VY=V0*SIN(ARG1)
      VZ=(VZ0*V0/VHOR0)*COS(ARG1)
C
      AX=-(VX0*V0/VHOR0)*OMEGA*SIN(ARG1)
      AY=V0*OMEGA*COS(ARG1)
      AZ=-(VZ0*V0/VHOR0)*OMEGA*SIN(ARG1)
C
   50 RHORSQ=X**2+Z**2
      RNGSQ=RHORSQ+Y**2
      RHOR=SQRT(RHORSQ)
      RANGE=SQRT(RNGSQ)
      FACT=X*VX+Z*VZ
      FACT1=FACT+Y*VY
      RHDOT=FACT/RHOR
      RNGDOT=FACT1/RANGE
      RHDDOT=(RHOR*(X*AX+Z*AZ+VX*VX+VZ*VZ)-FACT*RHDOT)/RHORSQ
C
      UT(1)=(X*VZ-Z*VX)/(RHORSQ*.00002)
      UT(2)=(RHOR*VY-Y*((X*VX+Z*VZ)/RHOR))/(RNGSQ*.00002)
C
      UT(3)=((X*AZ-AX*Z)-2.*FACT*(X*VZ-VX*Z)/RHORSQ)/(RHORSQ*.00002)
      UT(4)=((RHOR*AY-Y*RHDDOT)-2.*(VY*RHOR-Y*RHDOT)*RNGDOT/RANGE)/
     #      (RNGSQ*0.00002)
C
      RETURN
      END
      SUBROUTINE CROSS(A,B,PROD)
      REAL A(3),B(3),PROD(3)
      REAL MAG
C
C     THIS ROUTINE COMPUTES THE NORMALIZED CROSS PRODUCT OF TWO VECTO
C
      PROD(1)=A(2)*B(3)-A(3)*B(2)
      PROD(2)=-(A(1)*B(3)-A(3)*B(1))
      PROD(3)=A(1)*B(2)-A(2)*B(1)
C
      MAG=SQRT(PROD(1)**2+PROD(2)**2+PROD(3)**2)
C
      IF (MAG.EQ.1) GO TO 20
      DO 10 I=1,3
      PROD(I)=PROD(I)/MAG
   10 CONTINUE
   20 RETURN
      END
      SUBROUTINE DOT(A,B,PROD)
      REAL A(3),B(3)
C
C     THIS ROUTINE PRODUCES THE SCALAR:  A DOT B
C
      PROD=0.
      DO 10 I=1,3
      PROD=PROD+A(I)*B(I)
   10 CONTINUE
      RETURN
      END
```

```
          SUBROUTINE INITFRM(EVO,EPVO,EPPVO,VX,VY,VZ)
          REAL EVO(3),EPVO(3),EPPVO(3),J(3)
C
C         THIS ROUTINE COMPUTES THE INITIAL UNIT VECTORS OF THE TARGET
C         REFERENCE FRAME AT THE ONSET OF SOME MANEUVER.  IT SHOULD BE
C         CALLED WHEN A ROLL OR G-PULLING TURN IS INITATED.  IT SHOULD B
C         SUPPLIED WITH THE INERTIAL VELOCITY COMPONENTS AT THE TIME OF
C         MANEUVER INITIATION...(ASSUMES A CONST. SPEED TURN IF PULLING
C
C
          VMAG=SQRT(VX**2+VY**2+VZ**2)
          VHOR=SQRT(VX**2+VZ**2)
          EVO(1)=VX/VMAG
          EVO(2)=VY/VMAG
          EVO(3)=VZ/VMAG
C
C         EPVO IS THE RESULT OF NORMALIZED CROSS PRODUCT OF EVO AND J
C
          J(1)=J(3)=0.
          J(2)=1.
          CALL CROSS(EVO,J,EPVO)
C
C         EPPVO IS THE RESULT OF CROSSING EPVO AND EVO ... (NOTE ORDER)
C
          CALL CROSS(EPVO,EVO,EPPVO)
          WRITE(6,200) EVO,EPVO,EPPVO
200       FORMAT(/,1X,*TARGET REF. FRAME VECTORS AT MANEUVER INITIATION*
         #      /,1X,*EVO:*,T10,3F12.7,/,1X,*EPVO:*,T10,3F12.7,/,1X,*EPP
         #      ,T10,3F12.7)
          RETURN
          END
```

```
      SUBROUTINE LOCATE(XMAX,YMAX,SIGV,SIGPV,YT,UT,S,RANGE,
     #    X,Y,Z,VX,VY,VZ,FRAME,T0,T1,REVRT,DELV,DELPV,SIGVO,SIGPVO,NG,
     #    EVO,EPVO,EV,EPV,RF,NS)
      REAL XMAX(3),YMAX(3),SIGV(3),SIGPV(3),YT(2,1),UT(4)
      REAL DELV(3),DELPV(3),SIGVO(3),SIGPVO(3)
      REAL S(12),A(2,2),AT(2,2),P(2,2),PP(2,2),TEMP(2,2)
      REAL EA(3),EB(3),ER(3)
      REAL EVO(3),EPVO(3),EPPVO(3),EV(3),EPV(3),EPPV(3),RF(2,3)
      INTEGER FRAME
C
C
C     THIS ROUTINE IS USED TO LOCATE THE HOT SPOTS ON THE FLIR IMAGE
C     PLANE WHEN A DYNAMIC IMAGE IS BEING TRACKED.  THE CAUSSIAN DISP
C     PARAMETERS ARE ALSO COMPUTED ACCORDINGLY FROM THE REFERENCE
C     VALUES SPECIFIED FROM THE INPUT TO THE PROGRAM.
C
C
      DT=1./30.
      PI=3.141592654
C     DEFINE REFERENCE RANGE RELATED TO SIGVO AND SIGPVO
      REFRNG=20000.
      TIME=FLOAT(FRAME)/30.
      VMAG=SQRT(VX*VX+VY*VY+VZ*VZ)
      VPL=SQRT(UT(1)*UT(1)+UT(2)*UT(2))
      VMAX=VMAG/(RANGE*0.00002)
C
      SNTH=UT(2)/VPL
      CSTH=UT(1)/VPL
C
      DO 100 K=1,3
      SIGPV(K)=SIGPVO(K)*REFRNG/RANGE
      SIGV(K)=SIGPV(K)*(1.+(SIGVO(K)/SIGPVO(K)-1.)*VPL/VMAX)
  100 CONTINUE
C
C     DEFINE ALPHA, BETA, AND RANGE UNIT VECTORS
C
      ALPHA=ATAN(Z/X)
      BETA=ATAN(Y/SQRT(X*X+Z*Z))
C
      EA(1)=-SIN(ALPHA)
      EA(2)=0.
      EA(3)=COS(ALPHA)
C
      EB(1)=-COS(ALPHA)*SIN(BETA)
      EB(2)=COS(BETA)
      EB(3)=-SIN(ALPHA)*SIN(BETA)
C
      ER(1)=X/RANGE
      ER(2)=Y/RANGE
      ER(3)=Z/RANGE
C
      IF (TIME.GE.T1) GO TO 210
C
C     DEFINE EV,EPV,AND EPPV IN TERMS OF INIT VECTORS EVO,EPVO,AND EPP
C
C     GAMMA IS ANGLF OF ROLL FROM INITIAL ORIENTATION
C
```

```
      IF ((TIME-T0).LE.DT) CALL INITFRM(EVO,EPVO,EPPVO,VX,VY,VZ)
C
  120 GAMMA=2.*PI*REVRT*(TIME-T0)
      DO 200 K=1,3
      EV(K)=EVO(K)
      EPV(K)=COS(GAMMA)*EPVO(K)-SIN(GAMMA)*EPPVO(K)
C     EPPV(K)=SIN(GAMMA)*EPVO(K)+COS(GAMMA)*EPPVO(K)
  200 CONTINUE
  210 IF (TIME.LT.T1) GO TO 260
C
C     DEFINE EV,EPV,EPPV IN TERMS OF FRAME AT ONSET OF G-PULLING TURN
C     GAMMA IS ANGLE OF TURN FROM INIT POSITION
C
      IF ((TIME-T1).LE.DT) CALL INITFRM(EVO,EPVO,EPPVO,VX,VY,VZ)
C
      GAMMA=9.8*NG*(TIME-T1)/VMAG
      DO 250 K=1,3
      EV(K)=COS(GAMMA)*EVO(K)+SIN(GAMMA)*EPPVO(K)
      EPV(K)=EPVO(K)
C     EPPV(K)=-SIN(GAMMA)*EVO(K)+COS(GAMMA)*EPPVO(K)
  250 CONTINUE
C
C     COMPUTE REQUIRED DOT PRODUCTS TO PROJECT A VECTOR FROM TARGET
C     FRAME TO ALPHA-BETA PLANE
C
  260 CALL DOT(EA,EV,AVDOT)
      CALL DOT(EA,EPV,APVDOT)
      CALL DOT(EB,EV,BVDOT)
      CALL DOT(EB,EPV,BPVDOT)
      CALL DOT(ER,EV,RVDOT)
      CALL DOT(ER,EPV,RPVDOT)
C
      DO 300 K=1,3
C
C     DETERMINE LOCATIONS OF GAUSSIAN PEAKS BASED ON THE TRUE
C     CENTROID POSITION, YT
C
      XMAX(K)=YT(1,1)+(DELV(K)*AVDOT+DELPV(K)*APVDOT)*(REFRNG/RANGE)
      YMAX(K)=YT(2,1)+(DELV(K)*BVDOT+DELPV(K)*BPVDOT)*(REFRNG/RANGE)
C
C     CALCULATE AND LABLE DATA TO BE USED TO DETERMINE THE REALTIVE R
C     RANGES OF THE TARGET ELLIPSOIDS
C
C     FIRST ROW OF RF ARRAY CONTAINS ELLIPSOID LABELS
C     SECOND ROW CONTAINS RELATIVE RANGES
C
      RF(1,K)=FLOAT(K)
      RF(2,K)=DELV(K)*RVDOT+DELPV(K)*RPVDOT
  300 CONTINUE
C
C     NOW ORDER THE RANGES SUCH THAT RF(1,1) CONTAINS THE NAME OF THE
C     CLOSEST ELLIPSOID, ETC.
C
      DO 350 K=1,3
      IF (RF(2,K).GE.RF(2,1)) GO TO 320
      TEMP1=RF(1,1)
      TEMP2=RF(2,1)
      RF(1,1)=RF(1,K)
      RF(2,1)=RF(2,K)
      RF(1,K)=TEMP1
      RF(2,K)=TEMP2
320   IF (RF(2,K).LE.RF(2,3)) GO TO 350
      TEMP1=RF(1,3)
      TEMP2=RF(2,3)
      RF(1,3)=RF(1,K)
      RF(2,3)=RF(2,K)
      RF(1,K)=TEMP1
      RF(2,K)=TEMP2
350   CONTINUE
C
C     P - MATRIX WHOSE EIGENVALUES ARE THE INVERSE COVARIANCES OF THE
C         BIVARIATE GAUSSIAN INTENSITY DISTRIBUTIONS
C
C     PP - MATRIX WITH THE SAME EIGENVALUES BUT NOT DIAGONAL IN GENER
C     S - 12 ELEMENT ARRAY THAT WILL CONTAIN ALL VALUES OF THE THREE
C         2X2 PP MATRICES
C     A - DIRECTION COSINE MATRIX TO MAKE TRANSFORMATION TO FLIR COOR
C
      A(1,1)=CSTH
      A(1,2)=-SNTH
      A(2,1)=SNTH
      A(2,2)=CSTH
C
      DO 400 I=1,2
      DO 400 J=1,2
      AT(I,J)=A(J,I)
      P(I,J)=PP(I,J)=0.
400   CONTINUE
C
      L=0
      DO 500 K=1,3
      P(1,1)=1./SIGV(K)**2
      P(2,2)=1./SIGPV(K)**2
      CALL MULT(A,P,2,2,2,TEMP)
      CALL MULT(TEMP,AT,2,2,2,PP)
      DO 500 I=1,2
      DO 500 J=1,2
      L=L+1
      S(L)=PP(I,J)
500   CONTINUE
      GO TO 600
      IF (NS.EQ.1) PRINT *, "INERTIAL LOCATION= ",X,Y,Z
      IF (NS.EQ.1) PRINT *, "VPL=",VPL," VMAX=",VMAX
      IF (NS.EQ.1) PRINT *, "RANGE=",RANGE,"   GAMMA=",GAMMA
      IF (NS.EQ.1) PRINT *, "SIGV=",SIGV," SIGPV=",SIGPV
      IF (NS.EQ.1) PRINT *, "EA=",EA," EB=",EB
      IF (NS.EQ.1) PRINT *, "ER=",ER
      IF (NS.EQ.1) PRINT *, "EV=",EV," EPV=",EPV
      IF (NS.EQ.1) PRINT *, "A=",((A(I,J),J=1,2),I=1,2)
600   CONTINUE
      IF (NS.EQ.1) PRINT *, "RF=",RF
      RETURN
      END
      SUBROUTINE INITF(TAF,VARDF,VARAF,VARYQ,VARDFO)
C
C
C     THIS ROUTINE CONTROLS INPUTING VALUES NEEDED FOR THE KALMAN
C       - FILTER
C
C
      TAF=.07072
C
C     THIS IS THE CORRELATION TIME FOR THE ATMOSPHERIC MODEL FOR THE
C          - FILTER
C
C
C     THE VARIANCE OF THE DYNAMICS FOR THE FILTER
C
C
1     FORMAT(1X,*VARIANCE OF FILTER DYNAMICS = *,F14.4)
      READ(5,2) VARDFO
      WRITE(6,1) VARDFO
3     FORMAT(1X,*VARIANCE OF FILTER ATMOSPHERICS = *,F10.4)
      PRINT *, "(A NEGATIVE SIGN IMPLIES ADAPTIVE QFD)"
      VARDF=ABS(VARDFO)
      VARYQ=VARDF
      READ(5,2) VARAF
      WRITE(6,3) VARAF
2     FORMAT(F10.4)
C
C
C     THE VARIANCE OF THE ATMOSPHERIC JITTER FOR THE FILTER
C
C
      RETURN
      END
      SUBROUTINE FILTER(TDF,VARDF,TAF,VARAF,DT,PHIF,QFD,QFDMAX,QFDMIN)
      REAL PHIF(8,8),QFD(8,8),QFDMAX(8),QFDMIN(8)
C
C
C     THIS ROUTINE SETS UP THE STATE TRANSITION MATRIX AND QFD MATRIX
C
C
C     TAF   CORRELATION TIME FOR THE ATMOSPHERIC JITTER
C     TDF   CORRELATION TIME FOR THE TARGET DYNAMICS
C     VARDF TARGET DYNAMICS NOISE VARIANCE
C     VARAF ATMOSPHERIC NOISE VARIANCE
C
C
C     THE SOLUTION TO THE DYNAMIC EQUATIONS
C
C
C     XF(I+1)=PHIF*XF(I)
C
C
C     FOR PROPAGATION OF COVARIANCE MATRIX NEED QFD
C
C
      EX=EXP(-DT/TDF)
      EX2=EX*EX
      FACT=1.-EX
      TD2=TDF*TDF
      TD3=TDF*TD2
      TD4=TDF*TD3
      TDFDT=TDF*DT
      DT2=DT*DT
      DT3=DT*DT2
C
C     ZERO ALL MATRICIES
C
      DO 1 I=1,8
      DO 1 J=1,8
      PHIF(I,J)=0.
1     QFD(I,J)=0.
C
      PHIF(1,1)=1.
      PHIF(1,3)=DT
      PHIF(1,5)=TDF*(DT-TDF*FACT)
      PHIF(2,2)=1.
      PHIF(2,4)=DT
      PHIF(2,6)=PHIF(1,5)
      PHIF(3,3)=1.
      PHIF(3,5)=TDF*FACT
      PHIF(4,4)=1.
      PHIF(4,6)=PHIF(3,5)
      PHIF(5,5)=EXP(-DT/TDF)
      PHIF(6,6)=PHIF(5,5)
      PHIF(7,7)=EXP(-DT/TAF)
      PHIF(8,8)=PHIF(7,7)
C
C     FILL IN MAX QFD VALUES FOR QFD ESTIMATION.
C
      QFDMAX(1)=QFDMAX(2)=2.
      QFDMAX(3)=QFDMAX(4)=15.
      QFDMAX(5)=QFDMAX(6)=25.
      QFDMAX(7)=QFDMAX(8)=0.5
C
C     FILL IN MIN QFD VALUES FOR QFD ESTIMATION.
C
      QFDMIN(1)=QFDMIN(2)=0.1
      QFDMIN(3)=QFDMIN(4)=0.3
      QFDMIN(5)=QFDMIN(6)=4.0
      QFDMIN(7)=QFDMIN(8)=0.031
C
C
C
C
      QFD(1,1)=VARDF*(TD4+2.*(TD3*DT-TD2*DT2+TDF*DT3/3.)
     #         -TD3*EX*(4.*DT+TDF*EX))
      QFD(1,3)=VARDF*(TD3*(1.-2.*EX+EX2)-2.*TD2*DT*FACT+TDF*DT2)
      QFD(1,5)=VARDF*(TD2*(1.-EX2)-2.*TDFDT*EX)
      QFD(2,2)=QFD(1,1)
      QFD(2,4)=QFD(1,3)
      QFD(2,6)=QFD(1,5)
      QFD(3,1)=QFD(1,3)
      QFD(3,3)=VARDF*(TD2*(4.*EX-EX2-3.)+2.*TDFDT)
      QFD(3,5)=VARDF*TDF*FACT*FACT
      QFD(4,2)=QFD(2,4)
      QFD(4,4)=QFD(3,3)
      QFD(4,6)=QFD(3,5)
      QFD(5,1)=QFD(1,5)
      QFD(5,3)=QFD(3,5)
      QFD(5,5)=VARDF*(1.-EX2)
      QFD(6,2)=QFD(2,6)
      QFD(6,4)=QFD(4,6)
      QFD(6,6)=QFD(5,5)
      QFD(7,7)=VARAF*(1.-EXP(-2.*DT/TAF))
      QFD(8,8)=QFD(7,7)
C
C
      RETURN
      END
```

```
              SUBROUTINE PROPF(NR,NS,PHIF,QFD,PFP,PFM,XFP,XFM,TEMP2,MANIND)
              REAL PHIF(8,8),QFD(8,8),PFP(8,8),PFM(8,8),XFP(8),XFM(8)
              REAL TEMP1(8,8),TEMP2(8,8),PHIFT(8,8)
        C
        C
        C
        C     THIS ROUTINE IMPLEMENTS THE STATE TRANSITION
        C           -EQUATIONS FOR THE FILTER
        C
        C
        C     XFM(I+1)=PHIF*XFP(I)
        C
        C
        C     PFM=PHIF*PFP*PHIFT +QFD
        C
        C
        C     WHERE              PHIF=FILTER STATE TRANSITION MATRIX
        C                        XF  =FILTER STATE VECTOR
        C                        PFM =COV FILTER STATES MINUS
        C                       PFP =COV FILTER STATES PLUS
        C
        C
        C      PERFORM FILTER STATE PROPAGATION
        C
        C
              DO 1 I=1,8
              DO 1 J=1,8
        1     PHIFT(I,J)=PHIF(J,I)
              CALL MULT(PHIF,XFP,8,8,1,XFM)
              CALL MULT(PHIF,PFP,8,8,8,TEMP1)
              CALL MULT(TEMP1,PHIFT,8,8,8,TEMP2)
              DO 2 I=1,8
              DO 2 J=1,8
        2     PFM(I,J)=TEMP2(I,J)+QFD(I,J)
        C
              DO 30 I=1,8
              IF (I.LE.6) PFP(I,I)=AMAX1(PFP(I,I),(100.**((I+1)/2-2)))
              IF (PFM(I,I).GT.0.) GO TO 30
              PRINT *," NR =",NR," NS =",NS," PFM(",I,", ",I,") =",PFM(I,I)
              PFM(I,I)=ABS(PFM(I,I))
        30    CONTINUE
        C
              IF (NS.NE.1) GO TO 10
              WRITE (6,7300)
              WRITE(6,7420) ((QFD(J,J),J=1,8))
        7420  FORMAT(1X,*QFD*,1X,(8G14.5))
              WRITE(6,7422) ((PFP(J,J),J=1,8))
        7422  FORMAT(1X,*PFP*,1X,(8G14.5))
              IF ((10.LT.NR.AND.NR.LT.60).OR.(90.LT.NR)) GO TO 10
        7300  FORMAT(/,1X,*DIAGONAL ELEMENTS OF:* )
        C
              WRITE(6,7423) ((PFM(J,J),J=1,8))
        7423  FORMAT(1X,*PFM*,1X,(8G14.5))
              IF (MANIND.EQ.1) GO TO 10
              WRITE(6,7419) ((TEMP2(J,J),J=1,8))
        7419  FORMAT(1X,*PUPD*,1X,(8G14.5))
              WRITE(6,7421) ((PHIF(J,J),J=1,8))
```

```
      7421  FORMAT(1X,*PHIF*,1X,(8G14.5))
      10    RETURN
            END
      SUBROUTINE UPDAT(Z,LINH,NLINH,XFP,XFM,PFP,PFM,UPD,RINV,NS,NR,HTR
      REAL Z(64),LINH(64,8),NLINH(64),LINHT(8,64),XFP(8),XFM(8)
      REAL PFP(8,8),PFM(8,8),RINV(64,64),TEMP1(8,64),TEMP2(8,8)
      REAL XINV(8,8),OLDPP(8),HTRRES(8),HTRR(2)
      REAL RESID(64),PINV(8,8),GAINK(8,64),UPD(8)
C
C
C     THIS ROUTINE PROCESSES ONE MEASUREMENT VECTOR Z
C     AND ALSO UPDATES THE FILTER STATES XFP
C     Z      IS THE MEASUREMENT VECTOR OF FLIR MEASUREMENTS
C     NLINH  IS THE NONLINEAR INTENSITY FUNCTION AFTER SMOOTHING
C     LINH   IS THE LINEAR INTENSITY FUNCTION
C     XFP     IS FILTER STATES PLUS
C     XFM     IS FILTER STATES MINUS
C     PFP    COV MATRIX OF FILTER STATES PLUS
C     PFM     COV MATRIX OF FILTER STATES MINUS
C     RINV   INVERSE OF RFIL, THE FILTER MODEL OF SPATIALLY
C                 CORRELATED NOISE
C     TEMP1 WILL HOLD H-TRANSPOSE*RINV
C     PINV  WILL HOLD COV MATRIX MINUS INVERSED
C     TEMP2  WILL HOLD HTRINVH
C     GAINK  KALMAN FILTER GAIN
C     RESID  RESIDUAL
C     UPD     GAIN TIMES RESIDUAL
C     OLDPP - ARRAY THAT CONTAINS THE DIAG ELEMENTS OF OLD PFP MATRIX
C     HTRRES - ARRAY THAT HOLDS HT*RINV*RESID
C
C
C    THE EQUATIONS USED FOR THE UPDATE ARE:
C
C
C
C     FIRST FOR COMPUTING THE COVARINACE PLUS MATRIX
C     USING THE INVERSE COV METHOD
C
C
C     PINV(I+)=PINV(I-)+LINHT(I)*RINV(I)LINH(I)
C
C
C     PFP(I+)=[PINV(I+)!**-1
C
C
C     K(I)=P(I+)LINHT(I)RINV(I)
C
C
C     XFP(I)=XFM(I-)+K(I)[Z(I)-NLINH(I)!
C
C
C     FIRST CREAT LINH TRANSPOSED
C
C
      IF ((NR*NS).EQ.1) WRITE(6,8525) ((PFM(I,J),J=1,8),I=1,8)
8525  FORMAT(1X,*PFM*,/,(1X,8G14.5))
      DO 1 J=1,8
      OLDPP(J)=PFP(J,J)
      DO 1 I=1,64
1     LINHT(J,I)=LINH(I,J)
```

```
C
C     COMPUTE THE INVERSE OF THE COV PLUS MATRIX
      CALL MULT(LINHT,RINV,8,64,64,TEMP1)
      CALL MULT(TEMP1,LINH,8,64,8,TEMP2)
C
C     INVERT THE COVARIANCE MINUS MATRIX
      CALL INVERT(PFM,8,PINV)
      IF ((NR*NS).EQ.1) WRITE(6,8529) ((PINV(I,J),J=1,8),I=1,8)
8529  FORMAT(1X,*PINV*,/,(1X,8G14.5))
      IF ((NR*NS).EQ.1) WRITE(6,8530) ((TEMP2(I,J),J=1,8),I=1,8)
8530  FORMAT(1X,*TEMP2U*,/,(1X,8G14.5))
C
      DO 2 I=1,8
      DO 2 J=1,8
2     PINV(I,J)=PINV(I,J)+TEMP2(I,J)
C
C
C     CREATE P(I+)=PFP
      CALL INVERT(PINV,8,PFP)
C
      DO 30 I=1,8
      IF (I.GT.6) GO TO 29
C     BOUNDING OF P MATRIX REQUIRED WHEN QFD ESTIMATION PERFORMED
C     BOUND VALUES ARE 0.01, 0.1, AND 1.0 FOR THE POS, VEL, AND ACCEL
C     STATES
C
      BOUND=10.**((I+1)/2-3)
      FACT=SQRT(ABS(BOUND/PFP(I,I)))
      PFP(I,I)=AMAX1(PFP(I,I),BOUND)
      IF (FACT.LE.1.) GO TO 29
      DO 28 J=1,8
      IF (I.EQ.J) GO TO 28
      PFP(I,J)=PFP(I,J)*FACT
      PFP(J,I)=PFP(J,I)*FACT
28    CONTINUE
29    CONTINUE
      IF (PFP(I,I).GT.0.) GO TO 30
      PRINT *," NR =",NR," NS =",NS," PFP(",I,", ",I,") =",PFP(I,I)
      PFP(I,I)=ABS(PFP(I,I))
30    CONTINUE
C
C
      CALL  MULT(PINV,PFP,8,8,8,XINV)
      IF ((NR*NS).EQ.1) WRITE(6,7501) ((XINV(I,J),J=1,8),I=1,8)
7501  FORMAT(1X,*XINV*,/,(1X,8G14.5))
C
      IF ((NR*NS).EQ.1) WRITE(6,8420) ((PFP(I,J),J=1,8),I=1,8)
8420  FORMAT(1X,*PFPU*,/,(1X,8G14.5))
8424  FORMAT(1X,*LINHT*,/,(1X,8G14.5))
      IF ((NR*NS).EQ.1) WRITE(6,8426) ((RINV(I,J),J=1,64)
     #,I=1,2)
8426  FORMAT(1X,*R*,/,(1X,8G14.5))
8422  FORMAT(2X,*RINV HAD AT LEAST ONE NEGATIVE VALUE*)
C
C     COMPUTE THE KALMAN FILTER GAIN
      CALL MULT(PFP,TEMP1,8,8,64,GAINK)
      IF ((NR*NS).EQ.1) WRITE(6,8417) ((GAINK(I,J),J=1,64),I=1,2)
```

```
8417  FORMAT(1X,*GAINX*,/,8(1X,8G12.5,/),//,1X,*GAINY*,/,8(1X,8G12.5,
C
C     COMPUTE STATE MEASUREMENT UPDATE
      FIMAX=0.
      DO 3 I=1,64
      FIMAX=AMAX1(FIMAX,Z(I))
3     RESID(I)=Z(I)-NLINH(I)
      IF ((NR*NS).EQ.1) WRITE(6,8418) (RESID(I),I=1,64)
      IF ((NR*NS).EQ.1) WRITE(6,8423) (Z(I),I=1,64)
      IF ((NR*NS).EQ.1) WRITE(6,8444) (NLINH(I),I=1,64)
8423  FORMAT(1X,*Z*,/,(1X,8G14.5))
8444  FORMAT(1X,*NLINH*,/,(1X,8G14.5))
8418  FORMAT(1X,*RESID*,/,(1X,8G12.5))
      CALL MULT(TEMP1,RESID,8,64,1,HTRRES)
      IF (NS.EQ.1) CALL PRINT(HTRRES,1,8,10HHT*RI*RES )
      CALL MULT(PFP,HTRRES,8,8,1,UPD)
      IF ((NR*NS).EQ.1) WRITE(6,8419) UPD
8419  FORMAT(1X,*UPD*,8G12.5)
      DO 4 I=1,8
4     XFP(I)=XFM(I)+UPD(I)
C
C
      RETURN
      END
      SUBROUTINE QADAPT(FRAME,TRXXT,UPD,PUPD,QFD,QFDMIN,QFDMAX,
     #          PFP,VARDFO,TDF,VARAF,TAF,VARYQ,NS)
      INTEGER FRAME
      REAL UPD(8),PUPD(8,8),DXDXT(8,8),PFP(8,8)
      REAL QFD(8,8),QFD1(8,8),QFDMAX(8),QFDMIN(8)
C
C     THIS ROUTINE PERFORMS ADAPTIVE QFD ESTIMATION
C     THE INITIAL VALUE OF QFD IS REDUCED LINEARLY DURING
C     ACQUISITION.  THEN ESTIMATION BEGINS BY FRAME 15 AT THE LATEST.
C
C
      TRXXTO=TRXXT
      IF (FRAME.NE.1) TRXXT=0.
C
      DO 110 I=1,8
      DO 100 J=1,8
      DXDXT(I,J)=UPD(I)*UPD(J)
      QFD1(I,J)=QFD(I,J)
100   CONTINUE
      TRXXT=TRXXT+DXDXT(I,I)
110   CONTINUE
C
      TRXXT=0.8*TRXXTO+0.2*TRXXT
      IF (VARDFO.GT.0.) GO TO 74
C
      IF (FLOAT(FRAME).LT.6.) GO TO 200
      IF (FLOAT(FRAME).GE.15.) GO TO 250
      IF (TRXXT.GE.1000.) GO TO 250
      GO TO 74
C
C
C     ACQUISITION SCHEDULE CHANGE OF QFD
C
200   VARYQ=VARYQ+VARDFO*0.06444444444
      CALL QFDCOMP(QFD,VARYQ,TDF,VARAF,TAF)
      GO TO 74
C
C     ESTIMATION OF QFD
C
250   IF (NS.EQ.1) PRINT *," "
      IF (NS.EQ.1) PRINT *," QFD ADAPTATION AT FRAME ",FRAME
300   DO 400 I=1,8
      DO 400 J=1,8
400   QFD(I,J)=DXDXT(I,J)+PFP(I,J)-PUPD(I,J)
C
C     BOUND QFD TO PREVENT NEGATIVE EIGENVALUES
C
C        BOUNDING QFD
C
      DO 84 I=1,8
      QFACTOR=1.
      IF(QFD(I,I).GT.0.) GO TO 86
      QFACTOR=0.
      QFD(I,I) = AMAX1(0.1,(QFDMIN(I)**2))
      GO TO 85
 86   CONTINUE
      IF(SQRT(QFD(I,I)).GT.QFDMAX(I)) QFACTOR=QFDMAX(I)/SQRT(QFD(I,I
C
C     LOWER BOUND QFD
C
      IF (SQRT(QFD(I,I)).LT.QFDMIN(I)) QFACTOR=QFDMIN(I)/SQRT(QFD(I,I)
C
      IF (NS.EQ.1) PRINT *," QFACTOR NUMBER ",I," = ",QFACTOR
      IF(QFACTOR.EQ.1.) GO TO 84
      QFD(I,I)=QFACTOR**2*QFD(I,I)
 85   CONTINUE
      DO 82 J=1,8
      IF(I.EQ.J) GO TO 82
      QFD(I,J)=QFD(I,J)*QFACTOR
 82   CONTINUE
      DO 83 K=1,8
      IF(I.EQ.K) GO TO 83
      QFD(K,I)=QFD(K,I)*QFACTOR
 83   CONTINUE
 84   CONTINUE
      DO 73 I=1,8
      DO 73 J=1,8
 73   QFD(I,J)= .2*QFD(I,J)+.8*QFD1(I,J)
 74   CONTINUE
      RETURN
      END
      SUBROUTINE QFDCOMP(QFD,VARDF,TDF,VARAF,TAF)
      REAL QFD(8,8)
C
C     THIS ROUTINE COMPUTES CONSTANT QFD MATRICES THAT CORRESPOND TO
C     THE FIRST ORDER GAUSS-MARKOV MODEL.  THE ROUTINE IS REPEATEDLY
C     CALLED FOR QFD ADAPTATION DURING TARGET ACQUISITION.
C
      DT=1./30.
C
      EX=EXP(-DT/TDF)
      EX2=EX*EX
      FACT=1.-EX
      TD2=TDF*TDF
      TD3=TDF*TD2
      TD4=TDF*TD3
      TDFDT=TDF*DT
      DT2=DT*DT
      DT3=DT*DT2
C
C     ZERO QFD MATRIX
C
      DO 1 I=1,8
      DO 1 J=1,8
1     QFD(I,J)=0.
C
C
C
C
C
      QFD(1,1)=VARDF*(TD4+2.*(TD3*DT-TD2*DT2+TDF*DT3/3.)
     #       -TD3*EX*(4.*DT+TDF*EX))
      QFD(1,3)=VARDF*(TD3*(1.-2.*EX+EX2)-2.*TD2*DT*FACT+TDF*DT2)
      QFD(1,5)=VARDF*(TD2*(1.-EX2)-2.*TDFDT*EX)
      QFD(2,2)=QFD(1,1)
      QFD(2,4)=QFD(1,3)
      QFD(2,6)=QFD(1,5)
      QFD(3,1)=QFD(1,3)
      QFD(3,3)=VARDF*(TD2*(4.*EX-EX2-3.)+2.*TDFDT)
      QFD(3,5)=VARDF*TDF*FACT*FACT
      QFD(4,2)=QFD(2,4)
      QFD(4,4)=QFD(3,3)
      QFD(4,6)=QFD(3,5)
      QFD(5,1)=QFD(1,5)
      QFD(5,3)=QFD(3,5)
      QFD(5,5)=VARDF*(1.-EX2)
      QFD(6,2)=QFD(2,6)
      QFD(6,4)=QFD(4,6)
      QFD(6,6)=QFD(5,5)
      QFD(7,7)=VARAF*(1.-EXP(-2.*DT/TAF))
      QFD(8,8)=QFD(7,7)
C
C
      RETURN
      END
      SUBROUTINE INPUT3(IMAX,S,XMAX,YMAX,N,NZ,X,Y,DATA,CENX,CENY,
     #RF,NR,NS)
      REAL ARG(3),IMAX(3),ISIG(3),INTEN(3),S(12),XMAX(3),YMAX(3)
      REAL RF(2,3)
      INTEGER A(3),LOBE(24,24),OVERLAP
      COMPLEX DATA(N,N)
C       THIS ROUTINE DETERMINES REAL MODEL INTENSITY AND CENTROID
C     VALUES FOR AN 8X8 PIXEL FOV. ZERO PADDING IS ACCOMPLISHED BY
C     CENTERING THE 8X8 PIXEL FOV WITHIN A NULL NXN SPACE WHEN NZ>0.
C       THE COORDINATE SYSTEM IS AS DEFINED BELOW:
C
C
C         (0,0)........................
C              .                      .
C              .                      .
C              .          #           .
C              .     GAUSSIAN         .
C              .                      .
C              .                      .
C              .     (X,Y)........    .
C              .          . FOV  .    .
C              .     #    .     #.    .
C              .          ........    .
C              ........................
C
C
C     THE INTENSITY PATTERN IS DEFINED TO BE 3 GAUSSIAN DISTRIBUTIONS
C     OF INTENSITY IMAX(I),I=1,3 LOCATED AT XMAX(I),YMAX(I),I=1,3
C     WITH COVARIANCE S(I),I=1,3. THE UPPER LEFT CORNER OF THE 8X8 PI
C     FOV IS DEFINED TO BE LOCATION X,Y MICRORAD.
C      THE INTENSITY AT EACH PIXEL IS DETERMINED BY INTEGRATING THE
C     INTENSITY OF 25 EQUALLY SPACED SPOTS WITHIN THE PIXEL.
C
C
C
C     IMAX(I) - MAXIMUM INTENSITY OF THE ITH INTENSITY DISTRIBUTION
C     ISIG(I) - THE ONE SIGMA INTENSITY OF THE ITH DISTRIBUTION
C     INTEN(I) - CONTRIBUTION TO THE OVERALL INTENSITY AT A PARTICULA
C             POINT FROM THE ITH INTENSITY DISTRIBUTION
C
C     A(I) - FLAG INDICATING WHETHER  INTEN(I) >= ISIG(I)  1:YES, 2:N
C     RF(I) - THREE ELEMENT RANGE FLAG ARRAY:  THE FIRST ELEMENT
C             CONTAINS THE NUMBER OF THE DISTRIBUTION RELATED TO THE
C             TARGET ELLIPSOID CLOSEST TO THE INERTIAL ORIGIN SECOND
C             AND THIRD ELEMENTS APPLY SIMILARLY
C
C
C
C
C
C     ZERO OUT FOV SPACE
C
      DO 10 I=1,N
      DO 10 J=1,N
      LOBE(I,J)=0
10    DATA(I,J)=0.
      SUMX=0.
      SUMY =0.
      SUMAVG=0.
      IF(N.LT.8) N=8
      IF((N-8)/2.LT.NZ) NZ=(N-8)/2
      LM=NZ+1
      LP=N-NZ
      IB=(N-8)/2+1
      DO 1 I=LM,LP
      DO 2 J=LM,LP
      AVG=0.
C     DIVIDE PIXEL I,J INTO 25 SEGMENTS
      DO 3 K1=1,5
      DO 4 K2=1,5
      IF ((K1.NE.3.OR.K2.NE.3).AND.(I.LT.9.OR.I.GT.16 .OR. J.LT.9.OR.
     #J.GT.16)) GO TO 4
      DELY=(I-IB)*1.0+(K1-1)*.2
      DELX=(J-IB)*1.0+(K2-1)*.2
      XP=X+DELX
      YP=Y+DELY
      X1=XP-XMAX(1)
      Y 1 = Y P -Y M A X ( 1 )
      X2=XP-XMAX(2)
      Y2=YP-YMAX(2)
      X3=XP-XMAX(3)
      Y3=YP-YMAX(3)
      ARG(1)=-.5*((X1**2*S(1))+(X1*Y1)*(S(2)+S(3))+(Y1**2*S(4)))
      ARG(2)=-.5*((X2**2*S(5))+(X2*Y2)*(S(6)+S(7))+(Y2**2*S(8)))
      ARG(3)=-.5*((X3**2*S(9))+(X3*Y3)*(S(10)+S(11))+(Y3**2*S(12)))
C
C
C
      IFLAG=0
      FXY=0.
      NUMSIG=0
      DO 200 L=1,3
      IF (ARG(L).GT.-225.) GO TO 150
      ARG(L)=-225.
      IFLAG=1
150   ISIG(L)=0.60653066*IMAX(L)
      INTEN(L)=IMAX(L)*EXP(ARG(L))
      A(L)=0
      IF (INTEN(L).GE.ISIG(L)) A(L)=1
      NUMSIG=NUMSIG+A(L)
200   CONTINUE
C
      DO 300 L=1,3
C     IF NUMSIG=0 THEN ADD ALL THREE COMPONENTS TO FXY
      IF (NUMSIG.GT.0) GO TO 250
      FXY=FXY+INTEN(L)
250   IF (FXY.GT.0.) GO TO 300
      M=INT(RF(1,L))
C
C     FIND CLOSEST CONTRIBUTION WITHIN ONE SIGMA BOUND AND EQUATE FXY
      IF (A(M).EQ.1 .AND. FXY.EQ.0.) FXY=INTEN(M)
      IF (K1.EQ.3.AND.K2.EQ.3) LOBE(I,J)=M
300   CONTINUE
      IF (NUMSIG.EQ.0) FXY=FXY/3.
      C
      C
      C
            AVG=AVG+FXY
            SUMX=SUMX+XP*FXY
            SUMY=SUMY+YP*FXY
      4     CONTINUE
      3     CONTINUE
            DATA(I,J)=AVG
            IF(I.GE.9.OR.I.LE.16.OR.J.GE.9.OR.J.LE.16) DATA(I,J)=DATA(I,J)/
      C     WRITE(6,100) I,J,DATA(I,J)
      100   FORMAT(2X,2I4,2X,G12.5)
            SUMAVG=SUMAVG+AVG
      2     CONTINUE
      1     CONTINUE
            CENX=SUMX/SUMAVG
            CENY=SUMY/SUMAVG
            IF (NS.NE.1.OR.(NR/5)*5.NE.NR) GO TO 510
            PRINT *, "ELLIPSOID CONTRIBUTIONS"
            DO 505 K=1,8
            WRITE(6,500) (LOBE(17-K,J),J=9,16)
      500   FORMAT(T2,8I2)
      505   CONTINUE
      510   IF (NS*IFLAG.EQ.1) WRITE(6,555)
      555   FORMAT(1X,*INTENSITY DATA ARTIFICIALLY BOUNDED -- LOST TRACK*)
            RETURN
            END
      SUBROUTINE SINGLE(IMAX,S,XT,YT,XF,YF,DATA,DX,DY,NR,NS)
      COMPLEX DATA(24,24),DX(24,24),DY(24,24)
      REAL IMAX(3),S(12)
      INTEGER LOBE(24,24)
C
C     THIS ROUTINE IS USED TO GENERATE A SINGLE SPOT IMAGE
C     IT ONLY DETERMINES INTENSITY VALUES AT THE CENTER OF EACH PIXEL
C
C
      ITARG=0
      IFLAG=0
      DO 10 I=1,24
      DO 10 J=1,24
      LOBE(I,J)=0
      DX(I,J)=0.
      DY(I,J)=0.
10    DATA(I,J)=0.
      DO 100 I=1,24
      DO 100 J=1,24
      X1=XF-XT+FLOAT(J-12)-0.5
      Y1=YF-YT+FLOAT(I-12)-0.5
      ARG=(-.5*(X1**2*S(1)+(X1*Y1)*(S(2)+S(3))+
     # Y1**2*S(4)))
      IF (ARG.LT.-225.) IFLAG=1
      IF (ARG.LT.-225.) ARG=-225.
      DATA(I,J)=IMAX(1)*EXP(ARG)
      DX(I,J)=-(-X1*S(1)-.5*Y1*(S(2)+S(3)))*DATA(I,J)
      DY(I,J)=-(-Y1*S(4)-.5*X1*(S(2)+S(3)))*DATA(I,J)
      IF (DATA(I,J).GE.(.60653*IMAX(1))) LOBE(I,J)=1
100   CONTINUE
      DO 505 K=1,8
      IF (NS.EQ.1) WRITE(6,500) (LOBE(17-K,J),J=9,16)
500   FORMAT(* 1 SIGMA IMAGE*,8(T20,8I2),/)
505   CONTINUE
510   IF (NS*IFLAG.EQ.1) WRITE(6,555)
555   FORMAT(1X,*INTENSITY DATA ARTIFICIALLY BOUNDED -- LOST TRACK*)
      RETURN
      END
      SUBROUTINE IDEAL(IMAX,S,XMAX,YMAX,N,NZ,X,Y,DATA,DX,DY,RF,ITARG,
     #NS)
      REAL ISIG(3),INTEN(3),RF(2,3),ARG(9)
      REAL IMAX(3),XMAX(3),YMAX(3),S(12)
      INTEGER A(3),OVERLAP,LOBE(24,24)
      COMPLEX DATA(N,N),DX(N,N),DY(N,N)
C
C     THIS ROUTINE COMPUTES THE INTENSITY DATA FOR A THREE SPOT IMAGE
C     VALUES ARE COMPUTED AT THE CENTER OF EACH PIXEL.
C
      IFLAG=0
      IF (NS.EQ.1) PRINT *, "IN IDEAL:"
      IF (NS.EQ.1) PRINT *, "X=",X," XMAX(I)=",XMAX
      IF (NS.EQ.1) PRINT *, "Y=",Y," YMAX(I)=",YMAX
      IF (NS.EQ.1) PRINT *, "S= ",(S(I),I=1,4)
      IF (NS.EQ.1) PRINT *, "   ",(S(I),I=5,8)
      IF (NS.EQ.1) PRINT *, "   ",(S(I),I=9,12)
C
      DO 10 I=1,N
      DO 10 J=1,N
      DATA(I,J)=0.
      DX(I,J)=0.
      DY(I,J)=0.
      LOBE(I,J)=0
10    CONTINUE
      IF(N.LT.8) N=8
      IF((N-8)/2.LT.NZ) NZ=(N-8)/2
      LM=NZ+1
      LP=N-NZ
      IB=(N-8)/2+1
      DO 1 I=LM,LP
      DO 1 J=LM,LP
      X1=X+FLOAT(J-IB)-XMAX(1)+.5
      X2=X+FLOAT(J-IB)-XMAX(2)+.5
      X3=X+FLOAT(J-IB)-XMAX(3)+.5
      Y1=Y+FLOAT(I-IB)-YMAX(1)+.5
      Y2=Y+FLOAT(I-IB)-YMAX(2)+.5
      Y3=Y+FLOAT(I-IB)-YMAX(3)+.5
      ARG(1)=-.5*((X1**2*S(1))+(X1*Y1)*(S(2)+S(3))+(Y1**2*S(4)))
      ARG(2)=-.5*((X2**2*S(5))+(X2*Y2)*(S(6)+S(7))+(Y2**2*S(8)))
      ARG(3)=-.5*((X3**2*S(9))+(X3*Y3)*(S(10)+S(11))+(Y3**2*S(12)))
C
      ARG(4)=-X1*S(1)-.5*Y1*(S(2)+S(3))
      ARG(5)=-X2*S(5)-.5*Y2*(S(6)+S(7))
      ARG(6)=-X3*S(9)-.5*Y3*(S(10)+S(11))
      ARG(7)=-Y1*S(4)-.5*X1*(S(2)+S(3))
      ARG(8)=-Y2*S(8)-.5*X2*(S(6)+S(7))
      ARG(9)=-Y3*S(12)-.5*X3*(S(10)+S(11))
C
C
C
      IFLAG=0
      DO 200 L=1,3
      IF (ARG(L).GT.-225.) GO TO 150
      ARG(L)=-225.
      IFLAG=1
150   ISIG(L)=0.60653066*IMAX(L)
```

```
      INTEN(L)=IMAX(L)*EXP(ARG(L))
      A(L)=0
      IF (INTEN(L).GE.ISIG(L).AND.ITARG.EQ.1) A(L)=1
200   CONTINUE
C
      OVERLAP=0
      DO 300 L=1,3
      IF (OVERLAP.EQ.1) GO TO 300
      M=INT(RF(1,L))
      OVERLAP=OVERLAP+A(M)
C     LOBE IS ARRAY INDICATING WHICH ELLIPSOID DATA CAME FROM
      IF (OVERLAP.EQ.1) LOBE(I,J)=M
      DATA(I,J)=DATA(I,J)+INTEN(M)
      DX(I,J)=DX(I,J)+ARG(M+3)*INTEN(M)
      DY(I,J)=DY(I,J)+ARG(M+6)*INTEN(M)
300   CONTINUE
      IF (OVERLAP.NE.0) GO TO 1
C    IF NOT WITHIN ONE SIGMA BOUND OF ANY ELLIPSE THEN USE AVERAGE IN
C
      DATA(I,J)=DATA(I,J)/3.
      DX(I,J)=DX(I,J)/3.
      DY(I,J)=DY(I,J)/3.
1     CONTINUE
      IF (NS.EQ.1 .AND.((NR/5)*5).EQ.NR) WRITE(6,5679) ((LOBE(I,J),
     # J=8,17),I=8,17)
5679  FORMAT(* ELLIPSOID*,10(T15,10I2,/))
C
      IF ((IFLAG.EQ.1).AND.(NS.EQ.1)) WRITE(6,555)
555   FORMAT(/,1X,*INTENSITY DATA ARTIFICIALLY BOUNDED IN *IDEAL**)
      RETURN
      END
      SUBROUTINE SMOOTH(DATA,SDATA,ALPHA,N,ITERAT)
      COMPLEX DATA(N,N),SDATA(N,N)
C     THIS ROUTINE SMOOTHS RAW DATA ARRAY DATA USING EXPONENTIAL
C     SMOOTHING. WEIGHTING FACTOR ALPHA IS USED TO GENERATE THE
C     SMOOTHED DATA IN ARRAY SDATA. THE PARAMETER ITERATION IS
C     USED TO DETERMINE THE WEIGHTING FACTOR WHEN FEWER THEN
C     1/ALPHA ITERATIONS HAVE BEEN DONE.
      A=1./ITERAT
      IF(A.LT.ALPHA) A=ALPHA
1      DO  3  I=1,N
      DO 3 J=1,N
3     SDATA(I,J)=A*DATA(I,J)+(1.-A)*SDATA(I,J)
      RETURN
      END
      SUBROUTINE SPTN(SIGMAB,R,M)
      DIMENSION R(64,64),C(5)
      DATA C/.3679,.2431,.1353,.1069,.0591/
C
C     SET UP SPATIAL NOISE CORRELATION COEFFICIENT MATRIX
C     USING SECOND NEAREST NEIGHBOR CORRELATION.
C
C
C     C IS THE ARRAY CONTAINING THE NON-ZERO ELEMENTS
C     CORRESPONDING TO THE DISTANCES TO NEIGHBORING PIXELS.
C     THE ARRAY VALUES ARE EXP(-DISTANCE IN PIXELS)
C
C     SIGMAB IS THE BACKGROUND VARIANCE
C
C     R IS THE M**2 BY M**2 CORRELATION MATRIX
C
      N=M**2
      DO 30 I=1,N
      DO 30 J=1,N
      R(I,J)=0.0
30    CONTINUE
      DO 36 I=1,N
      R(I,I)=1.
      IF (I.GE.64) GO TO 36
      R(I,I+1)=C(1)
      IF (I.GE.63) GO TO 36
      R(I,I+2)=C(3)
      IF(I.GE.59) GO TO 36
      R(I,I+6)=C(4)
      IF(I.GE.58) GO TO 36
      R(I,I+7)=C(2)
      IF(I.GE.57) GO TO 36
      R(I,I+8)=C(1)
      IF(I.GE.56) GO TO 36
      R(I,I+9)=C(2)
      IF(I.GE.55) GO TO 36
      R(I,I+10)=C(4)
      IF(I.GE.51) GO TO 36
      R(I,I+14)=C(5)
      IF(I.GE.50) GO TO 36
      R(I,I+15)=C(4)
      IF(I.GE.49) GO TO 36
      R(I,I+16)=C(3)
      IF(I.GE.48) GO TO 36
      R(I,I+17)=C(4)
      IF(I.GE.47) GO TO 36
      R(I,I+18)=C(5)
36    CONTINUE
      DO 37 I=1,M
      R(8*I-7,8*I)=0.0
      R(8*I-7,8*I-1)=0.0
      R(8*I-6,8*I)=0.0
      IF (I.GE.8) GO TO 37
      R(8*I,8*I+1)=0.0
      R(8*I,8*I+2)=0.0
      R(8*I-1,8*I+1)=0.0
      R(8*I-7,8*I+7)=0.0
      R(8*I-7,8*I+8)=0.0
      R(8*I-6,8*I+8)=0.0
      IF (I.GE.7) GO TO 37
      R(8*I,8*I+9)=0.0
      R(8*I,8*I+10)=0.0
      R(8*I-1,8*I+9)=0.0
      IF (I.GE.6) GO TO 37
      R(8*I,8*I+17)=0.0
      R(8*I,8*I+18)=0.0
      R(8*I-1,8*I+17)=0.0
37    CONTINUE
      DO 38 I=1,N
      L=I+1
      DO 38 J=L,N
      IF(L.GT.N) GO TO 38
      R(J,I)=R(I,J)
38    CONTINUE
      DO 39 I=1,N
      DO 39 J=1,N
39    R(I,J)=SIGMAB*R(I,J)
      RETURN
      END
```

```
          SUBROUTINE NOISE(W,N)
          REAL W(N)
          IA=1
    1     DO 2 I=1,N
          CALL GAUSS(IA,IY,VAL)
          W(I)=VAL
          IA=IY
    2     CONTINUE
          RETURN
          END
```

```
      SUBROUTINE DISPLAY(IXSIZE,IYSIZE,DATA)
      INTEGER IXY(122)
      COMPLEX DATA(IYSIZE,IXSIZE)
      WRITE(6,102)
102   FORMAT(1H1)
      NUM=IXSIZE+2
      DO 5 I=1,NUM
5     IXY(I)=1H-
      WRITE(6,103) (IXY(I),I=1,NUM)
103   FORMAT(T4,122A1)
      DMIN=1.E30
      DMAX=-1.E30
      DO 1 I=1,IXSIZE
      DO 1 J=1,IYSIZE
      DMAX=AMAX1(DMAX,CABS(DATA(J,I)))
1     DMIN =AMIN1 (DMIN ,CABS(DATA(J,I)))
      DO 4 J=1,IYSIZE
      DO 2 I=1,IXSIZE
      IXY(I)=1H
      X=(CABS(DATA(J,I))-DMIN)/(DMAX-DMIN)
      IF(X.GT..42) IXY(I)=1HO
      IF((X.GT..28).AND.(X.LT..42)) IXY(I)=1HX
      IF ((X.GT..14).AND.(X.LT..28)) IXY(I)=1H+
2     CONTINUE
      NUM=IXSIZE+1
      IXY(NUM)=1H!
      WRITE(6,100) (IXY(I),I=1,NUM)
100   FORMAT(T4,*!*,121A1)
      DO 3 I=1,IXSIZE
      IXY(I)=1H
      X=(CABS(DATA(J,I))-DMIN)/(DMAX-DMIN)
      IF((X.GT..56).AND.(X.LE..70)) IXY(I)=1H-
      IF((X.GT..70).AND.(X.LE..84)) IXY(I)=1H+
      IF(X.GT..84) IXY(I)=1H#
3     CONTINUE
      WRITE(6,101) (IXY(I),I=1,IXSIZE)
101   FORMAT(1H+,T5,120A1)
4     CONTINUE
      NUM=IXSIZE+2
      DO 6 I=1,NUM
6     IXY(I)=1H-
      WRITE(6,103) (IXY(I),I=1,NUM)
      RETURN
      END
      SUBROUTINE SHIFT(DATA,N,XSHIFT,YSHIFT)
      COMPLEX DATA(N,N),TEMP1,TEMP2,TEMP3,TEMP4,FX,FXC,FY,FYC
C      THIS ROUTINE IMPLEMENTS A SPATIAL PHASE SHIFT
C     IN THE FREQUENCY DOMAIN. THE ARRAY DATA IS ASSUMED TO BE THE
C     NXN ARRAY OF FOURIER TRANSFORM COMPONENTS AS GENERATED BY THE
C     TRANSFORM ROUTINE FOURT. FOR EXAMPLE, FOR A 6X6 ARRAY DATA
C
C
C           -----------------------------------------------
C         [ X0 Y0 [ X1 Y0 [ X2 Y0 [ X3 Y0 [X2* Y0 [X1* Y0 [
C           -----------------------------------------------
C         [ X0 Y1 [ X1 Y1 [ X2 Y1 [ X3 Y1 [X2* Y1 [X1* Y1 [
C           -----------------------------------------------
C         [ X0 Y2 [ X1 Y2 [ X3 Y2 [ X3 Y2 [X2* Y2 [X1* Y2 [
C           -----------------------------------------------
C         [ X0 Y3 [ X1 Y3 [ X2 Y3 [ X3 Y3 [X2* Y3 [X1* Y3 [
C           -----------------------------------------------
C         [ X0 Y2*[ X1 Y2*[ X2 Y2*[ X3 Y2*[X2* Y2*[X1* Y2*[
C           -----------------------------------------------
C         [ X0 Y1*[ X1 Y1*[ X2 Y1*[ X3 Y1*[X2* Y1*[X1* Y1*[
C            ------------------------------------------------
C
C            PHASE SHIFTING IS IMPLEMENTED BY MULTIPLYING THE
C         FOURIER TRANSFORM COMPONENTS BY
C                   EXP(J*2*PI(FX*XSHIFT+FY*YSHIFT))
C
C         XSHIFT AND YSHIFT ARE THE SHIFTS IN THE X AND Y COORDINATE
C             DIRECTIONS.
C
C
7     PI=3.141592654
      DEM=FLOAT(N)
      NCENT=N/2+1
      DO 1 I=1,NCENT
      DO 1 J=1,NCENT
      FX=CMPLX(0.,-2.*PI*(J-1)*XSHIFT/DEM)
      FY=CMPLX(0.,-2.*PI*(I-1)*YSHIFT/DEM)
      FXC=CONJG(FX)
      FYC=CONJG(FY)
      TEMP1=DATA(I,J)
      DATA(I,J)=TEMP1*CEXP(FX+FY)
      IF(I.EQ.1) GO TO 10
      IF(J.EQ.1.OR.J.EQ.NCENT) GO TO 20
      TEMP2=DATA(I,N+2-J)
      TEMP3=DATA(N+2-I,J)
      TEMP4=DATA(N+2-I,N+2-J)
      DATA(I,N+2-J)=TEMP2*CEXP(FXC+FY)
      DATA(N+2-I,J)=TEMP3*CEXP(FX+FYC)
      DATA(N+2-I,N+2-J)=TEMP4*CEXP(FXC+FYC)
      GO TO 1
10    IF(J.EQ.1.OR.J.EQ.NCENT) GO TO 1
      TEMP2=DATA(I,N+2-J)
      DATA(I,N+2-J)=TEMP2*CEXP(FXC+FY)
      GO TO 1
20    TEMP3=DATA(N+2-I,J)
      DATA(N+2-I,J)=TEMP3*CEXP(FX+FYC)
1     CONTINUE
      RETURN
```

```
          END
```

```
      SUBROUTINE DERIV(DATA,N,DX,DY)
      COMPLEX DATA(N,N),TEMP1,TEMP2,TEMP3,TEMP4,FX,FXC,FY,FYC
      COMPLEX DX(N,N),DY(N,N)
C      THIS ROUTINE IMPLEMENTS SPATIAL PARTIAL DERIVATIVES
C     IN THE FREQUENCY DOMAIN. THE ARRAY DATA IS ASSUMED TO BE THE
C     NXN ARRAY OF FOURIER TRANSFORM COMPONENTS AS GENERATED BY THE
C     TRANSFORM ROUTINE FOURT. FOR EXAMPLE, FOR A 6X6 ARRAY DATA
C
C         -------------------------------------------------
C         [ X0 Y0 [ X1 Y0 [ X2 Y0 [ X3 Y0 [X2* Y0 [X1* Y0 [
C         -------------------------------------------------
C         [ X0 Y1 [ X1 Y1 [ X2 Y1 [ X3 Y1 [X2* Y1 [X1* Y1 [
C         -------------------------------------------------
C         [ X0 Y2 [ X1 Y2 [ X3 Y2 [ X3 Y2 [X2* Y2 [X1* Y2 [
C         -------------------------------------------------
C         [ X0 Y3 [ X1 Y3 [ X2 Y3 [ X3 Y3 [X2* Y3 [X1* Y3 [
C         ------------------------ ------------------------
C         [ X0 Y2*[ X1 Y2*[ X2 Y2*[ X3 Y2*[X2* Y2*[X1* Y2*[
C         -------------------------------------------------
C         [ X0 Y1*[ X1 Y1*[ X2 Y1*[ X3 Y1*[X2* Y1*[X1* Y1*[
C         -------------------------------------------------
C
C            DIFFERENTIATION IS IMPLEMENTED BY MULTIPLYING THE
C            FOURIER TRANSFORM COMPONENTS BY
C                       J*2*PI*FX  AND  J*2*PI*FY
C
C
7     PI=3.141592654
      DEM=FLOAT(N)
      NCENT=N/2+1
      DO 1 I=1,NCENT
      DO 1 J=1,NCENT
      FX=CMPLX(0.,+2.*PI*(J-1)/DEM)
      FY=CMPLX(0.,+2.*PI*(I-1)/DEM)
      FXC=CONJG(FX)
      FYC=CONJG(FY)
      TEMP1=DATA(I,J)
      DX(I,J)=TEMP1*FX
      DY(I,J)=TEMP1*FY
      IF(I.EQ.1) GO TO 10
      IF(J.EQ.1.OR.J.EQ.NCENT) GO TO 20
      TEMP2=DATA(I,N+2-J)
      TEMP3=DATA(N+2-I,J)
      TEMP4=DATA(N+2-I,N+2-J)
      DX(I,N+2-J)=TEMP2*FXC
      DY(I,N+2-J)=TEMP2*FY
      DX(N+2-I,J)=TEMP3*FX
      DY(N+2-I,J)=TEMP3*FYC
      DX(N+2-I,N+2-J)=TEMP4*FXC
      DY(N+2-I,N+2-J)=TEMP4*FYC
      GO TO 1
10    IF(J.EQ.1.OR.J.EQ.NCENT) GO TO 1
      TEMP2=DATA(I,N+2-J)
      DX(I,N+2-J)=TEMP2*FXC
      DY(I,N+2-J)=TEMP2*FY
      GO TO 1
20    TEMP3=DATA(N+2-I,J)
```

```
                    DX(N+2-I,J)=TEMP3*FX
                    DY(N+2-I,J)=TEMP3*FYC
60            1     CONTINUE
                    RETURN
                    END
```

```
      SUBROUTINE FOURT(DATA,NN,NDIM,ISIGN,IFORM,WORK)
C     FOR INFORMATION CONTACT MR. MARK HALLER 4950/ADDS/56248
C
C     THE COOLEY-TUKEY FAST FOURIER TRANSFORM IN USASI BASIC FORTRAN
C
C
C     TRANSFORM(K1,K2,...) = SUM(DATA(J1,J2,...)*EXP(ISIGN*2*PI*SQRT(
C     *((J1-1)*(K1-1)/NN(1)+(J2-1)*(K2-1)/NN(2)+...))), SUMMED FOR AL
C     J1, K1 BETWEEN 1 AND NN(1), J2, K2 BETWEEN 1 AND NN(2), ETC.
C     THERE IS NO LIMIT TO THE NUMBER OF SUBSCRIPTS.  DATA IS A
C     MULTIDIMENSIONAL COMPLEX ARRAY WHOSE REAL AND IMAGINARY
C     PARTS ARE ADJACENT IN STORAGE, SUCH AS FORTRAN IV PLACES THEM.
C     IF ALL IMAGINARY PARTS ARE ZERO (DATA ARE DISGUISED REAL), SET
C     IFORM TO ZERO TO CUT THE RUNNING TIME BY UP TO FORTY PERCENT.
C     OTHERWISE, IFORM = +1.  THE LENGTHS OF ALL DIMENSIONS ARE
C     STORED IN ARRAY NN, OF LENGTH NDIM.  THEY MAY BE ANY POSITIVE
C     INTEGERS, THO THE PROGRAM RUNS FASTER ON COMPOSITE INTEGERS, AN
C     ESPECIALLY FAST ON NUMBERS RICH IN FACTORS OF TWO.  ISIGN IS +1
C     OR -1.  IF A -1 TRANSFORM IS FOLLOWED BY A +1 ONE (OR A +1
C     BY A -1) THE ORIGINAL DATA REAPPEAR, MULTIPLIED BY NTOT (=NN(1)
C     NN(2)*...).  TRANSFORM VALUES ARE ALWAYS COMPLEX, AND ARE RETUR
C     IN ARRAY DATA, REPLACING THE INPUT.  IN ADDITION, IF ALL
C     DIMENSIONS ARE NOT POWERS OF TWO, ARRAY WORK MUST BE SUPPLIED,
C     COMPLEX OF LENGTH EQUAL TO THE LARGEST NON 2**K DIMENSION.
C     OTHERWISE, REPLACE WORK BY ZERO IN THE CALLING SEQUENCE.
C     NORMAL FORTRAN DATA ORDERING IS EXPECTED, FIRST SUBSCRIPT VARYI
C     FASTEST.  ALL SUBSCRIPTS BEGIN AT ONE.
C
C     RUNNING TIME IS MUCH SHORTER THAN THE NAIVE NTOT**2, BEING
C     GIVEN BY THE FOLLOWING FORMULA.  DECOMPOSE NTOT INTO
C     2**K2 * 3**K3 * 5**K5 * ....  LET SUM2 = 2*K2, SUMF = 3*K3 + 5*
C     + ... AND NF = K3 + K5 + ....  THE TIME TAKEN BY A MULTI-
C     DIMENSIONAL TRANSFORM ON THESE NTOT DATA IS T = T0 + NTOT*(T1+
C     T2*SUM2+T3*SUMF+T4*NF).  ON THE CDC 3300 (FLOATING POINT ADD TI
C     OF SIX MICROSECONDS), T = 3000 + NTOT*(500+43*SUM2+68*SUMF+
C     320*NF) MICROSECONDS ON COMPLEX DATA.  IN ADDITION, THE
C     ACCURACY IS GREATLY IMPROVED, AS THE RMS RELATIVE ERROR IS
C     BOUNDED BY 3*2**(-B)*SUM(FACTOR(J)**1.5), WHERE B IS THE NUMBER
C     OF BITS IN THE FLOATING POINT FRACTION AND FACTOR(J) ARE THE
C     PRIME FACTORS OF NTOT.
C
C     PROGRAM BY NORMAN BRENNER FROM THE BASIC PROGRAM BY CHARLES
C     RADER.  RALPH ALTER SUGGESTED THE IDEA FOR THE DIGIT REVERSAL.
C     MIT LINCOLN LABORATORY, AUGUST 1967.  THIS IS THE FASTEST AND M
C     VERSATILE VERSION OF THE FFT KNOWN TO THE AUTHOR.  SHORTER PRO-
C     GRAMS FOUR1 AND FOUR2 RESTRICT DIMENSION LENGTHS TO POWERS OF T
C     SEE-- IEEE AUDIO TRANSACTIONS (JUNE 1967), SPECIAL ISSUE ON FFT
C
C     THE DISCRETE FOURIER TRANSFORM PLACES THREE RESTRICTIONS UPON T
C     DATA.
C     1.  THE NUMBER OF INPUT DATA AND THE NUMBER OF TRANSFORM VALUES
C     MUST BE THE SAME.
C     2.  BOTH THE INPUT DATA AND THE TRANSFORM VALUES MUST REPRESENT
C     EQUISPACED POINTS IN THEIR RESPECTIVE DOMAINS OF TIME AND
C     FREQUENCY.  CALLING THESE SPACINGS DELTAT AND DELTAF, IT MUST B
C     TRUE THAT DELTAF=2*PI/(NN(I)*DELTAT).  OF COURSE, DELTAT NEED N
C     BE THE SAME FOR EVERY DIMENSION.
```

```
C     3. CONCEPTUALLY AT LEAST, THE INPUT DATA AND THE TRANSFORM OUTP
C     REPRESENT SINGLE CYCLES OF PERIODIC FUNCTIONS.
C
C     EXAMPLE 1.  THREE-DIMENSIONAL FORWARD FOURIER TRANSFORM OF A
C     COMPLEX ARRAY DIMENSIONED 32 BY 25 BY 13 IN FORTRAN IV.
C     DIMENSION DATA(32,25,13),WORK(50),NN(3)
C     COMPLEX DATA
C     DATA NN/32,25,13/
C     DO 1 I=1,32
C     DO 1 J=1,25
C     DO 1 K=1,13
C   1 DATA(I,J,K)=COMPLEX VALUE
C     CALL FOURT(DATA,NN,3,-1,1,WORK)
C
C     EXAMPLE 2.  ONE-DIMENSIONAL FORWARD TRANSFORM OF A REAL ARRAY O
C     LENGTH 64 IN FORTRAN II.
C     DIMENSION DATA(2,64)
C     DO 2 I=1,64
C     DATA(1,I)=REAL PART
C   2 DATA(2,I)=0.
C     CALL FOURT(DATA,64,1,-1,0,0)
C
      DIMENSION DATA(1),NN(1),IFACT(32),WORK(1)
C     CDC 6600 INITIALIZATION
C     WRITE(6,9000)
      WR=0.
      WI=0.
      WSTPR=0.
      WSTPI=0.
      TWOPI=6.283185307
      IF(NDIM-1)920,1,1
1     NTOT=2
      DO 2 IDIM=1,NDIM
      IF(NN(IDIM))920,920,2
2     NTOT=NTOT*NN(IDIM)
C
C     MAIN LOOP FOR EACH DIMENSION
C
      NP1=2
      DO 910 IDIM=1,NDIM
      N=NN(IDIM)
      NP2=NP1*N
      IF(N-1)920,900,5
C
C     FACTOR N
C
5     M=N
      NTWO=NP1
      IF=1
      IDIV=2
10    IQUOT=M/IDIV
      IREM=M-IDIV*IQUOT
      IF(IQUOT-IDIV)50,11,11
11    IF(IREM)20,12,20
12    NTWO=NTWO+NTWO
      M=IQUOT
      GO TO 10
      20    IDIV=3
      30    IQUOT=M/IDIV
            IREM=M-IDIV*IQUOT
            IF(IQUOT-IDIV)60,31,31
      31    IF(IREM)40,32,40
      32    IFACT(IF)=IDIV
            IF=IF+1
            M=IQUOT
            GO TO 30
      40    IDIV=IDIV+2
            GO TO 30
      50    IF(IREM)60,51,60
      51    NTWO=NTWO+NTWO
            GO TO 70
      60    IFACT(IF)=M
      C
      C     SEPARATE FOUR CASES--
      C      1. COMPLEX TRANSFORM OR REAL TRANSFORM FOR THE 4TH, 5TH,ETC.
      C         DIMENSIONS.
      C       2. REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION.  METHOD--
      C         TRANSFORM HALF THE DATA, SUPPLYING THE OTHER HALF BY CON-
      C         JUGATE SYMMETRY.
      C      3. REAL TRANSFORM FOR THE 1ST DIMENSION, N ODD.  METHOD--
      C         TRANSFORM HALF THE DATA AT EACH STAGE, SUPPLYING THE OTHER
      C         HALF BY CONJUGATE SYMMETRY.
      C       4. REAL TRANSFORM FOR THE 1ST DIMENSION, N EVEN.  METHOD--
      C         TRANSFORM A COMPLEX ARRAY OF LENGTH N/2 WHOSE REAL PARTS
      C         ARE THE EVEN NUMBERED REAL VALUES AND WHOSE IMAGINARY PARTS
      C         ARE THE ODD NUMBERED REAL VALUES.  SEPARATE AND SUPPLY
      C         THE SECOND HALF BY CONJUGATE SYMMETRY.
      C
      70    NON2=NP1*(NP2/NTWO)
            ICASE=1
            IF(IDIM-4)71,90,90
      71    IF(IFORM)72,72,90
      72    ICASE=2
            IF(IDIM-1)73,73,90
      73    ICASE=3
            IF(NTWO-NP1)90,90,74
      74    ICASE=4
            NTWO=NTWO/2
            N=N/2
            NP2=NP2/2
            NTOT=NTOT/2
            I=3
            DO 80 J=2,NTOT
            DATA(J)=DATA(I)
      80    I=I+2
      90    I1RNG=NP1
            IF(ICASE-2)100,95,100
      95    I1RNG=NP0*(1+NPREV/2)
      C
      C     SHUFFLE ON THE FACTORS OF TWO IN N.  AS THE SHUFFLING
      C     CAN BE DONE BY SIMPLE INTERCHANGE, NO WORKING ARRAY IS NEEDED
      C
      100   IF(NTWO-NP1)600,600,110
      110   NP2HF=NP2/2
      J=1
      DO 150 I2=1,NP2,NON2
      IF(J-I2)120,130,130
  120 I1MAX=I2+NON2-2
      DO 125 I1=I2,I1MAX,2
      DO 125 I3=I1,NTOT,NP2
      J3=J+I3-I2
      TEMPR=DATA(I3)
      TEMPI=DATA(I3+1)
      DATA(I3)=DATA(J3)
      DATA(I3+1)=DATA(J3+1)
      DATA(J3)=TEMPR
  125 DATA(J3+1)=TEMPI
  130 M=NP2HF
  140 IF(J-M)150,150,145
  145 J=J-M
      M=M/2
      IF(M-NON2)150,140,140
  150 J=J+M
C
C     MAIN LOOP FOR FACTORS OF TWO.  PERFORM FOURIER TRANSFORMS OF
C     LENGTH FOUR, WITH ONE OF LENGTH TWO IF NEEDED.  THE TWIDDLE FA
C     W=EXP(ISIGN*2*PI*SQRT(-1)*M/(4*MMAX)).  CHECK FOR W=ISIGN*SQRT
C     AND REPEAT FOR W=ISIGN*SQRT(-1)*CONJUGATE(W).
C
      NON2T=NON2+NON2
      IPAR=NTWO/NP1
  310 IF(IPAR-2)350,330,320
  320 IPAR=IPAR/4
      GO TO 310
  330 DO 340 I1=1,I1RNG,2
      DO 340 J3=I1,NON2,NP1
      DO 340 K1=J3,NTOT,NON2T
      K2=K1+NON2
      TEMPR=DATA(K2)
      TEMPI=DATA(K2+1)
      DATA(K2)=DATA(K1)-TEMPR
      DATA(K2+1)=DATA(K1+1)-TEMPI
      DATA(K1)=DATA(K1)+TEMPR
  340 DATA(K1+1)=DATA(K1+1)+TEMPI
  350 MMAX=NON2
  360 IF(MMAX-NP2HF)370,600,600
  370 LMAX=MAX0(NON2T,MMAX/2)
      IF(MMAX-NON2)405,405,380
  380 THETA=-TWOPI*FLOAT(NON2)/FLOAT(4*MMAX)
      IF(ISIGN)400,390,390
  390 THETA=-THETA
  400 WR=COS(THETA)
      WI=SIN(THETA)
      WSTPR=-2.*WI*WI
      WSTPI=2.*WR*WI
  405 DO 570 L=NON2,LMAX,NON2T
      M=L
      IF(MMAX-NON2)420,420,410
  410 W2R=WR*WR-WI*WI
      W2I=2.*WR*WI
      W3R=W2R*WR-W2I*WI
```

```
          W3I=W2R*WI+W2I*WR
   420    DO 530 I1=1,I1RNG,2
          DO 530 J3=I1,NON2,NP1
          KMIN=J3+IPAR*M
          IF(MMAX-NON2)430,430,440
   430    KMIN=J3
   440    KDIF=IPAR*MMAX
   450    KSTEP=4*KDIF
          DO 520 K1=KMIN,NTOT,KSTEP
          K2=K1+KDIF
          K3=K2+KDIF
          K4=K3+KDIF
          IF(MMAX-NON2)460,460,480
   460    U1R=DATA(K1)+DATA(K2)
          U1I=DATA(K1+1)+DATA(K2+1)
          U2R=DATA(K3)+DATA(K4)
          U2I=DATA(K3+1)+DATA(K4+1)
          U3R=DATA(K1)-DATA(K2)
          U3I=DATA(K1+1)-DATA(K2+1)
          IF(ISIGN)470,475,475
   470    U4R=DATA(K3+1)-DATA(K4+1)
          U4I=DATA(K4)-DATA(K3)
          GO TO 510
   475    U4R=DATA(K4+1)-DATA(K3+1)
          U4I=DATA(K3)-DATA(K4)
          GO TO 510
   480    T2R=W2R*DATA(K2)-W2I*DATA(K2+1)
          T2I=W2R*DATA(K2+1)+W2I*DATA(K2)
          T3R=WR*DATA(K3)-WI*DATA(K3+1)
          T3I=WR*DATA(K3+1)+WI*DATA(K3)
          T4R=W3R*DATA(K4)-W3I*DATA(K4+1)
          T4I=W3R*DATA(K4+1)+W3I*DATA(K4)
          U1R=DATA(K1)+T2R
          U1I=DATA(K1+1)+T2I
          U2R=T3R+T4R
          U2I=T3I+T4I
          U3R=DATA(K1)-T2R
          U3I=DATA(K1+1)-T2I
          IF(ISIGN)490,500,500
   490    U4R=T3I-T4I
          U4I=T4R-T3R
          GO TO 510
   500    U4R=T4I-T3I
          U4I=T3R-T4R
   510    DATA(K1)=U1R+U2R
          DATA(K1+1)=U1I+U2I
          DATA(K2)=U3R+U4R
          DATA(K2+1)=U3I+U4I
          DATA(K3)=U1R-U2R
          DATA(K3+1)=U1I-U2I
          DATA(K4)=U3R-U4R
   520    DATA(K4+1)=U3I-U4I
          KMIN=4*(KMIN-J3)+J3
          KDIF=KSTEP
          KDIF=KSTEP
          IF(KDIF-NP2)450,530,530
   530    CONTINUE
```

```
      M=MMAX-M
      IF(ISIGN)540,550,550
  540 TEMPR=WR
      WR=-WI
      WI=-TEMPR
      GO TO 560
  550 TEMPR=WR
      WR=WI
      WI=TEMPR
  560 IF(M-LMAX)565,565,410
  565 TEMPR=WR
      WR=WR*WSTPR-WI*WSTPI+WR
  570 WI=WI*WSTPR+TEMPR*WSTPI+WI
      IPAR=3-IPAR
      MMAX=MMAX+MMAX
      GO TO 360
C
C     MAIN LOOP FOR FACTORS NOT EQUAL TO TWO.  APPLY THE TWIDDLE FACT
C     W=EXP(ISIGN*2*PI*SQRT(-1)*(J2-1)*(J1-J2)/(NP2*IFP1)), THEN
C     PERFORM A FOURIER TRANSFORM OF LENGTH IFACT(IF), MAKING USE OF
C     CONJUGATE SYMMETRIES.
C
  600 IF(NTWO-NP2)605,700,700
  605 IFP1=NON2
      IF=1
      NP1HF=NP1/2
  610 IFP2=IFP1/IFACT(IF)
      J1RNG=NP2
      IF(ICASE-3)612,611,612
  611 J1RNG=(NP2+IFP1)/2
      J2STP=NP2/IFACT(IF)
      J1RG2=(J2STP+IFP2)/2
  612 J2MIN=1+IFP2
      IF(IFP1-NP2)615,640,640
  615 DO 635 J2=J2MIN,IFP1,IFP2
      THETA=-TWOPI*FLOAT(J2-1)/FLOAT(NP2)
      IF(ISIGN)625,620,620
  620 THETA=-THETA
  625 SINTH=SIN(THETA/2.)
      WSTPR=-2.*SINTH*SINTH
      WSTPI=SIN(THETA)
      WR=WSTPR+1.
      WI=WSTPI
      J1MIN=J2+IFP1
      DO 635 J1=J1MIN,J1RNG,IFP1
      I1MAX=J1+I1RNG-2
      DO 630 I1=J1,I1MAX,2
      DO 630 I3=I1,NTOT,NP2
      J3MAX=I3+IFP2-NP1
      DO 630 J3=I3,J3MAX,NP1
      TEMPR=DATA(J3)
      DATA(J3)=DATA(J3)*WR-DATA(J3+1)*WI
  630 DATA(J3+1)=TEMPR*WI+DATA(J3+1)*WR
      TEMPR=WR
      WR=WR*WSTPR-WI*WSTPI+WR
  635 WI=TEMPR*WSTPI+WI*WSTPR+WI
  640 THETA=-TWOPI/FLOAT(IFACT(IF))
```

```
      IF(ISIGN)650,645,645
  645 THETA=-THETA
  650 SINTH=SIN(THETA/2.)
      WSTPR=-2.*SINTH*SINTH
      WSTPI=SIN(THETA)
      KSTEP=2*N/IFACT(IF)
      KRANG=KSTEP*(IFACT(IF)/2)+1
      DO 698 I1=1,I1RNG,2
      DO 698 I3=I1,NTOT,NP2
      DO 690 KMIN=1,KRANG,KSTEP
      J1MAX=I3+J1RNG-IFP1
      DO 680 J1=I3,J1MAX,IFP1
      J3MAX=J1+IFP2-NP1
      DO 680 J3=J1,J3MAX,NP1
      J2MAX=J3+IFP1-IFP2
      K=KMIN+(J3-J1+(J1-I3)/IFACT(IF))/NP1HF
      IF(KMIN-1)655,655,665
  655 SUMR=0.
      SUMI=0.
      DO 660 J2=J3,J2MAX,IFP2
      SUMR=SUMR+DATA(J2)
  660 SUMI=SUMI+DATA(J2+1)
      WORK(K)=SUMR
      WORK(K+1)=SUMI
      GO TO 680
  665 KCONJ=K+2*(N-KMIN+1)
      J2=J2MAX
      SUMR=DATA(J2)
      SUMI=DATA(J2+1)
      OLDSR=0.
      OLDSI=0.
      J2=J2-IFP2
  670 TEMPR=SUMR
      TEMPI=SUMI
      SUMR=TWOWR*SUMR-OLDSR+DATA(J2)
      SUMI=TWOWR*SUMI-OLDSI+DATA(J2+1)
      OLDSR=TEMPR
      OLDSI=TEMPI
      J2=J2-IFP2
      IF(J2-J3)675,675,670
  675 TEMPR=WR*SUMR-OLDSR+DATA(J2)
      TEMPI=WI*SUMI
      WORK(K)=TEMPR-TEMPI
      WORK(KCONJ)=TEMPR+TEMPI
      TEMPR=WR*SUMI-OLDSI+DATA(J2+1)
      TEMPI=WI*SUMR
      WORK(K+1)=TEMPR+TEMPI
      WORK(KCONJ+1)=TEMPR-TEMPI
  680 CONTINUE
      IF(KMIN-1)685,685,686
  685 WR=WSTPR+1.
      WI=WSTPI
      GO TO 690
  686 TEMPR=WR
      WR=WR*WSTPR-WI*WSTPI+WR
      WI=TEMPR*WSTPI+WI*WSTPR+WI
  690 TWOWR=WR+WR
```

```
      IF(ICASE-3)692,691,692
  691 IF(IFP1-NP2)695,692,692
  692 K=1
      I2MAX=I3+NP2-NP1
      DO 693 I2=I3,I2MAX,NP1
      DATA(I2)=WORK(K)
      DATA(I2+1)=WORK(K+1)
  693 K=K+2
      GO TO 698
C
C     COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N ODD, BY CON-
C     JUGATE SYMMETRIES AT EACH STAGE.
C
  695 J3MAX=I3+IFP2-NP1
      DO 697 J3=I3,J3MAX,NP1
      J2MAX=J3+NP2-J2STP
      DO 697 J2=J3,J2MAX,J2STP
      J1MAX=J2+J1RG2-IFP2
      J1CNJ=J3+J2MAX+J2STP-J2
      DO 697 J1=J2,J1MAX,IFP2
      K=1+J1-I3
      DATA(J1)=WORK(K)
      DATA(J1+1)=WORK(K+1)
      IF(J1-J2)697,697,696
  696 DATA(J1CNJ)=WORK(K)
      DATA(J1CNJ+1)=-WORK(K+1)
  697 J1CNJ=J1CNJ-IFP2
  698 CONTINUE
      IF=IF+1
      IFP1=IFP2
      IF(IFP1-NP1)700,700,610
C
C     COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N EVEN, BY CON
C     JUGATE SYMMETRIES.
C
  700 GO TO (900, 800, 900, 701) ICASE
  701 NHALF=N
      N=N+N
      THETA=-TWOPI/FLOAT(N)
      IF(ISIGN)703,702,702
  702 THETA=-THETA
  703 SINTH=SIN(THETA/2.)
      WSTPR=-2.*SINTH*SINTH
      WSTPI=SIN(THETA)
      WR=WSTPR+1.
      WI=WSTPI
      IMIN=3
      JMIN=2*NHALF-1
      GO TO 725
  710 J=JMIN
      DO 720 I=IMIN,NTOT,NP2
      SUMR=(DATA(I)+DATA(J))/2.
      SUMI=(DATA(I+1)+DATA(J+1))/2.
      DIFR=(DATA(I)-DATA(J))/2.
      DIFI=(DATA(I+1)-DATA(J+1))/2.
      TEMPR=WR*SUMI+WI*DIFR
      TEMPI=WI*SUMI-WR*DIFR
      DATA(I)=SUMR+TEMPR
      DATA(I+1)=DIFI+TEMPI
      DATA(J)=SUMR-TEMPR
      DATA(J+1)=-DIFI+TEMPI
  720 J=J+NP2
      IMIN=IMIN+2
      JMIN=JMIN-2
      TEMPR=WR
      WR=WR*WSTPR-WI*WSTPI+WR
      WI=TEMPR*WSTPI+WI*WSTPR+WI
  725 IF(IMIN-JMIN)710,730,740
  730 IF(ISIGN)731,740,740
  731 DO 735 I=IMIN,NTOT,NP2
  735 DATA(I+1)=-DATA(I+1)
  740 NP2=NP2+NP2
      NTOT=NTOT+NTOT
      J=NTOT+1
      IMAX=NTOT/2+1
  745 IMIN=IMAX-2*NHALF
      I=IMIN
      GO TO 755
  750 DATA(J)=DATA(I)
      DATA(J+1)=-DATA(I+1)
  755 I=I+2
      J=J-2
      IF(I-IMAX)750,760,760
  760 DATA(J)=DATA(IMIN)-DATA(IMIN+1)
      DATA(J+1)=0.
      IF(I-J)770,780,780
  765 DATA(J)=DATA(I)
      DATA(J+1)=DATA(I+1)
  770 I=I-2
      J=J-2
      IF(I-IMIN)775,775,765
  775 DATA(J)=DATA(IMIN)+DATA(IMIN+1)
      DATA(J+1)=0.
      IMAX=IMIN
      GO TO 745
  780 DATA(1)=DATA(1)+DATA(2)
      DATA(2)=0.
      GO TO 900
C
C     COMPLETE A REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION BY
C     CONJUGATE SYMMETRIES.
C
  800 IF(I1RNG-NP1)805,900,900
  805 DO 860 I3=1,NTOT,NP2
      I2MAX=I3+NP2-NP1
      DO 860 I2=I3,I2MAX,NP1
      IMIN=I2+I1RNG
      IMAX=I2+NP1-2
      JMAX=2*I3+NP1-IMIN
      IF(I2-I3)820,820,810
  810 JMAX=JMAX+NP2
  820 IF(IDIM-2)850,850,830
  830 J=JMAX+NP0
      DO 840 I=IMIN,IMAX,2
```

```
                    DATA(I)=DATA(J)
515                 DATA(I+1)=-DATA(J+1)
              840   J=J-2
              850   J=JMAX
                    DO 860 I=IMIN,IMAX,NP0
                    DATA(I)=DATA(J)
520                 DATA(I+1)=-DATA(J+1)
              860   J=J-NP0
              C
              C     END OF LOOP ON EACH DIMENSION
              C
525           900   NP0=NP1
                    NP1=NP2
              910   NPREV=N
              920   RETURN
                    END
```

```
NR. SEVERITY  DETAILS    DIAGNOSIS OF PROBLEM

495    I      DATA       ARRAY REFERENCE OUTSIDE DIMENSION BOUNDS.
496    I      DATA       ARRAY REFERENCE OUTSIDE DIMENSION BOUNDS.
```

```
      SUBROUTINE GAUSS(IA,IY,VAL)
C
C     THIS ROUTINE CALCULATES A GAUSSIAN DISTRIBUTED RANDOM VARIABLE
C     VAL, WITH MEAN=0. AND STANDARD DEVIATION=1.
C
C     IA IS INITIALIZED BEFORE FIRST CALL TO ANY ODD INTEGER LESS THI
C     10 DIGITS IN LENGTH.
C     IY IS GENERATED AND SHOULD BE USED FOR IA ON THE NEXT CALL TO
C     THIS ROUTINE.
C
      VAL=0.
      DO 1 I=1,12
      X=RANF(DUM)
      IA=IY
      VAL=VAL+X
    1 CONTINUE
      VAL=VAL-6.
      RETURN
      END
      SUBROUTINE INVERT(A,N,B,IER)
      IMPLICIT REAL (A-H,O-Z)
      DIMENSION A(1),B(1)
      REAL L(128),M(128)
      NSQ=N*N
      DO 1000 I=1,NSQ
 1000 B(I)=A(I)
      D=1.0
      NK=-N
      DO 80 K=1,N
      NK=NK+N
      L(K)=K
      M(K)=K
      KK=NK+K
      BIGA=A(KK)
      DO 20 J=K,N
      IZ=N*(J-1)
      DO 20 I=K,N
      IJ=IZ+I
   10 IF(ABS(BIGA)-ABS(A(IJ))) 15,20,20
   15 BIGA=A(IJ)
      L(K)=I
      M(K)=J
   20 CONTINUE
      J=L(K)
      IF(J-K) 35,35,25
   25 KI=K-N
      DO 30 I=1,N
      KI=KI+N
      HOLD=-A(KI)
      JI=KI-K+J
      A(KI)=A(JI)
   30 A(JI)=HOLD
   35 I=M(K)
      IF(I-K) 45,45,38
   38 JP=N*(I-1)
      DO 40 J=1,N
      JK=NK+J
      JI=JP+J
      HOLD=-A(JK)
      A(JK)=A(JI)
   40 A(JI)=HOLD
   45 IF(BIGA) 48,46,48
   46 D=0.0
      IER=129
      GO TO 150
   48 DO 55 I=1,N
      IF(I-K) 50,55,50
   50 IK=NK+I
      A(IK)=A(IK)/(-BIGA)
   55 CONTINUE
      DO 65 I=1,N
      IK=NK+I
      IJ=I-N
      DO 65 J=1,N
      IJ=IJ+N
      IF(I-K) 60,65,60
 60     IF(J-K) 62,65,62
 62     KJ=IJ-I+K
        A(IJ)=A(IK)*A(KJ)+A(IJ)
 65     CONTINUE
        KJ=K-N
        DO 75 J=1,N
        KJ=KJ+N
        IF(J-K) 70,75,70
 70     A(KJ)=A(KJ)/BIGA
 75     CONTINUE
        D=D*BIGA
        A(KK)=1.0/BIGA
 80     CONTINUE
        K=N
 100    K=(K-1)
        IF(K) 150,150,105
 105    I=L(K)
        IF(I-K) 120,120,108
 108    JQ=N*(K-1)
        JR=N*(I-1)
        DO 110 J=1,N
        JK=JQ+J
        HOLD=A(JK)
        JI=JR+J
        A(JK)=-A(JI)
 110    A(JI)=HOLD
 120    J=M(K)
        IF(J-K) 100,100,125
 125    KI=K-N
        DO 130 I=1,N
        KI=KI+N
        HOLD=A(KI)
        JI=KI-K+J
        A(KI)=-A(JI)
 130    A(JI)=HOLD
        GO TO 100
 150    DO 1002 I=1,NSQ
        SAVE=A(I)
        A(I)=B(I)
        B(I)=SAVE
 1002   CONTINUE
        RETURN
        END
```

```
      SUBROUTINE MULT(A,B,L,M,N,C)
      REAL A(L,M),B(M,N),C(L,N)
      DO 300 I=1,L
      DO 200 J=1,N
      C(I,J)=0.
      DO 100 INDEX=1,M
      C(I,J)=C(I,J)+A(I,INDEX)*B(INDEX,J)
  100 CONTINUE
  200 CONTINUE
  300 CONTINUE
      RETURN
      END
      SUBROUTINE CHOLY(A,N,S)
C
C     THIS ROUTINE DETERMINES THE LOWER TRIANGULAR CHOLESKY SQUARE-
C     ROOT OF AN NXN MATRIX.
C     A IS THE INPUT MATRIX AND S IS THE CHOLESKY SQUARE-ROOT MATRIX
C
      DIMENSION A(N,N),S(N,N)
      DO 120 I=1,N
      DO 120 J=1,N
120   S(I,J)=0.
      DO 123 I=1,N
      DO 123 J=1,N
      IF(ABS(A(I,J)).GT.1.E-06) GO TO 124
123   CONTINUE
      RETURN
124   S(1,1)=SQRT(A(1,1))
      DO 5 I=2,N
      IM1=I-1
      DO 3 J=1,IM1
      JM1=J-1
      SUM=0.
      DO 2 K=1,JM1
2     SUM=SUM+S(I,K)*S(J,K)
3     S(I,J)=(A(I,J)-SUM)/S(J,J)
      SUM=0.
      DO 4 K=1,IM1
4     SUM=SUM+S(I,K)**2
5     S(I,I)=SQRT(A(I,I)-SUM)
      RETURN
      END
      SUBROUTINE PRINT(A,NR,NC,NAME)
      DIMENSION A(NR,NC)
C
C     THIS ROUTINE PRINTS OUT MATRICIES
C
1     FORMAT(1X,//,2X,A15,* MATRIX*,/)
      WRITE (6,1) NAME
      DO 10 I=1,NR
      WRITE (6,5) (A(I,J), J=1,NC)
5     FORMAT(1X,8(F14.5,1X))
10    CONTINUE
      RETURN
      END
```

Replacement Subroutines

CTR Filter

```
      SUBROUTINE FILTER(TDF,VARDF,TAF,VARAF,DT,PHIF,QFD,QFDMAX,QFDMIN
      PEAL PHIF(8,8),QFD(8,8),QFDMAX(8),QFDMIN(8)
C
C
C     THIS ROUTINE SETS UP THE STATE TRANSITION MATRIX AND QFD MATRIX
C
C
C     TAF   CORRELATION TIME FOR THE ATMOSPHERIC JITTER
C     TDF   CORRELATION TIME FOR THE TARGET DYNAMICS
C     VARDF TARGET DYNAMICS NCISE VARIANCE
C     VARAF ATMOSPHERIC NCISE VAPIANCE
C
C     THE SOLUTION TO THE DYNAMIC EQUATIONS
C
C
C     XF(I+1)=PHIF*XF(I)
C
C
C     FOR PROPAGATION OF CCVARIANCE MATRIX NEED QFD
C
C
      EX=EXP(-DT/TDF)
      EX2=EX*EX
      FACT=1.-EX
      TD2=TDF*TDF
      TD3=TDF*TD2
      TD4=TDF*TD3
      TDFDT=TDF*DT
      DT2=DT*DT
      DT3=DT*DT2
C
C     ZERO ALL MATRICIES
C
      DO 1 I=1,8
      DO 1 J=1,8
      PHIF(I,J)=0.
1     QFD(I,J)=0.
C
C     DETERMINE PORTION OF PHI MATRIX THAT REMAINS CONSTANT
C
      PHIF(1,1)=1.
      PHIF(1,3)=DT
      PHIF(1,5)=.5*DT2
      PHIF(2,2)=1.
      PHIF(2,4)=DT
      PHIF(2,6)=PHIF(1,5)
      PHIF(3,3)=1.
      PHIF(3,5)=DT
      PHIF(4,4)=1.
      PHIF(4,6)=PHIF(3,5)
      PHIF(7,7)=EXP(-DT/TAF)
      PHIF(8,8)=PHIF(7,7)
C
C
C     FILL IN MAX QFD VALUES FOR QFD ESTIMATION.
C
      QFDMAX(1)=QFDMAX(2)=2.
```

```
                      QFDMAX(3)=QFDMAX(4)=15.
                      QFDMAX(5)=QFDMAX(6)=25.
  60                  QFDMAX(7)=QFDMAX(8)=0.5
                C
                C     FILL IN MIN QFD VALUES FOR QFD ESTIMATION.
                C
                      QFDMIN(1)=QFDMIN(2)=0.1
  65                  QFDMIN(3)=QFDMIN(4)=0.3
                      QFDMIN(5)=QFDMIN(6)=3.0
                      QFDMIN(7)=QFDMIN(8)=0.031
                C
                C     IF DESIRE QFD THAT IS THE SAME AS THE GAUSS-MARKOV MODEL THEN
  70            C     COMMENT THE NEXT STATEMENT
                      GO TO 100
                C
                C
                C
  75                  QFD(1,1)=VARDF*(TD4+2.*(TD3*DT-TD2*DT2+TDF*DT3/3.)
                     #           -TD3*EX*(4.*DT+TDF*EX))
                      QFD(1,3)=VARDF*(TD3*(1.-2.*EX+EX2)-2.*TD2*DT*FACT+TDF*DT2)
                      QFD(1,5)=VARDF*(TD2*(1.-EX2)-2.*TDFDT*EX)
                      QFD(2,2)=QFD(1,1)
  80                  QFD(2,4)=QFD(1,3)
                      QFD(2,6)=QFD(1,5)
                      QFD(3,1)=QFD(1,3)
                      QFD(3,3)=VARDF*(TD2*(4.*EX-EX2-3.)+2.*TDFDT)
                      QFD(3,5)=VARDF*TDF*FACT*FACT
  85                  QFD(4,2)=QFD(2,4)
                      QFD(4,4)=QFD(3,3)
                      QFD(4,6)=QFD(3,5)
                      QFD(5,1)=QFD(1,5)
                      QFD(5,3)=QFD(3,5)
  90                  QFD(6,2)=QFD(2,6)
                      QFD(6,4)=QFD(4,6)
                100   CONTINUE
                      QFD(5,5)=VARDF*(1.-EX2)
                      QFD(6,6)=QFD(5,5)
  95                  QFD(7,7)=VARAF*(1.-EXP(-2.*DT/TAF))
                      QFD(8,8)=QFD(7,7)
                C
                C
                      RETURN
 100                  END


NR. SEVERITY  DETAILS    DIAGNOSIS OF PROBLEM

 75     I                   THERE IS NO PATH TO THIS STATEMENT.
```

```
      SUBROUTINE PROPF(NR,NS,PHIF,QFD,PFP,PFM,XFP,XFM,TEMP2,MANIND)
      REAL PHIF(8,8),QFD(8,8),PFP(8,8),PFM(8,8),XFP(8),XFM(8)
      REAL TEMP1(8,8),TEMP2(8,8),PHIFT(8,8)
C
C
C
C     THIS ROUTINE IMPLEMENTS THE STATE TRANSITION
C          -EQUATIONS FOR THE FILTER
C
C
C     XFM(I+1)=PHIF*XFP(I)
C
C
C     PFM=PHIF*PFP*PHIFT +QFD
C
C
C     WHERE                PHIF=FILTER STATE TRANSITION MATRIX
C                         XF   =FILTER STATE VECTOR
C                         PFM  =COV FILTER STATES MINUS
C                        PFP  =COV FILTER STATES PLUS
C
C
C
C      PERFORM FILTER STATE PROPAGATION
C
C
C     FIRST DETERMINE THE CHANGES IN THE PHI MATRIX
C
C
      DT=1./30.
      DT2=DT*DT
      A1=XFP(3)**2+XFP(4)**2
      A2=XFP(3)*XFP(6)-XFP(4)*XFP(5)
      OMEGA=A2/A1
      WA1=OMEGA/A1
C
      PHIF(5,3)=-DT*WA1*(A2+2.*XFP(3)*(XFP(6)-2.*OMEGA*XFP(3)))
      PHIF(5,4)=2.*DT*WA1*XFP(3)*(XFP(5)+2.*XFP(4)*OMEGA)
      PHIF(5,5)=1.+2.*DT*WA1*XFP(3)*XFP(4)
      PHIF(5,6)=-2.*DT*WA1*XFP(3)*XFP(3)
      PHIF(6,3)=-2.*DT*WA1*XFP(4)*(XFP(6)-2.*XFP(3)*OMEGA)
      PHIF(6,4)=-DT*WA1*(A2-2.*XFP(4)*(XFP(5)+2.*XFP(4)*OMEGA))
      PHIF(6,5)=2.*DT*WA1*XFP(4)*XFP(4)
      PHIF(6,6)=1.-2.*DT*WA1*XFP(3)*XFP(4)
C
      DO 1 I=1,8
      DO 1 J=1,8
1     PHIFT(I,J)=PHIF(J,I)
C
      XFM(1)=XFP(1)+DT*XFP(3)+.5*DT2*XFP(5)
      XFM(2)=XFP(2)+DT*XFP(4)+.5*DT2*XFP(6)
      XFM(3)=XFP(3)+DT*XFP(5)
      XFM(4)=XFP(4)+DT*XFP(6)
      XFM(5)=XFP(5)-DT*OMEGA*OMEGA*XFP(3)
      XFM(6)=XFP(6)-DT*OMEGA*OMEGA*XFP(4)
      XFM(7)=XFP(7)*PHIF(7,7)
      XFM(8)=XFP(8)*PHIF(8,8)
```

```
C
      CALL MULT(PHIF,PFP,8,8,8,TEMP1)
      CALL MULT(TEMP1,PHIFT,8,8,8,TEMP2)
      DO 2 I=1,8
      DO 2 J=1,8
2     PFM(I,J)=TEMP2(I,J)+QFD(I,J)
C
      DO 30 I=1,8
      IF (PFM(I,I).GT.0.) GO TO 30
      PRINT *," NR =",NR," NS =",NS," PFM(",I,", ",I,") =",PFM(I,I)
      PFM(I,I)=ABS(PFM(I,I))
30    CONTINUE
C
      IF (NS.NE.1) GO TO 10
      WRITE (6,7300)
      WRITE(6,7420) ((QFD(J,J),J=1,8))
7420  FORMAT(1X,*QFD*,1X,(8E14.5))
      WRITE(6,7422) ((PFP(J,J),J=1,8))
7422  FORMAT(1X,*PFP*,1X,(8E14.5))
      IF ((10.LT.NR.AND.NR.LT.60).OR.(80.LT.NR)) GO TO 10
7300  FORMAT(/,1X,*DIAGONAL ELEMENTS OF:* )
C
      WRITE(6,7423) ((PFM(J,J),J=1,8))
7423  FORMAT(1X,*PFM*,1X,(8E14.5))
      IF (MANIND.EQ.1) GO TO 10
      WRITE(6,7419) ((TEMP2(J,J),J=1,8))
7419  FORMAT(1X,*PUPD*,1X,(8E14.5))
      WRITE(6,7421) ((PHIF(6,J),J=1,8))
      WRITE(6,7421) ((PHIF(5,J),J=1,8))
7421  FORMAT(1X,*PHIF*,1X,(8E14.5))
10    RETURN
      END
```

Plotting Routine

```
      PROGRAM HFZ(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE8,TAPE9)
      REAL TIME(155),SIG(155),RMS(155),PP(155),PM(155)
      REAL EST(155),ESTMS(155),ESTPS(155),RANGE(5)
      REAL MPEAK,MEANAV
      INTEGER NAME(20,4)
      CALL PLOTS
C
C     READ IN REQUIRED DATA
C
      REWIND 8
      READ(8,100) NG,NFRMS,NRUNS,ITARG
      READ(8,200) ALPHA,SIGAT,VARM,VARDFO
100   FORMAT(5I3)
200   FORMAT(6E12.5)
300   FORMAT(3F15.6)
400   FORMAT(55X,3F15.6)
C
C     ASSIGN PLOT TITLES TO CORRESPONDING DATA SETS
C
      NAME(1,1)=10HX MINUS PO
      NAME(2,1)=10HY MINUS PO
      NAME(3,1)=10HX MINUS VE
      NAME(4,1)=10HY MINUS VE
      NAME(5,1)=10HX MINUS AC
      NAME(6,1)=10HY MINUS AC
      NAME(7,1)=10HX CEN MINU
      NAME(8,1)=10HY CEN MINU
      NAME(9,1)=10HX PLUS POS
      NAME(10,1)=10HY PLUS POS
      NAME(11,1)=10HX PLUS VEL
      NAME(12,1)=10HY PLUS VEL
      NAME(13,1)=10HX PLUS ACC
      NAME(14,1)=10HY PLUS ACC
      NAME(15,1)=10HX CEN PLUS
      NAME(16,1)=10HY CEN PLUS
      NAME(1,2)=NAME(2,2)=NAME(7,2)=NAME(8,2)=10HS
      NAME(3,2)=NAME(4,2)=10HL
      NAME(13,2)=NAME(14,2)=10HEL
      NAME(5,2)=NAME(6,2)=10HCEL
      NAME(15,2)=NAME(16,2)=NAME(9,2)=NAME(10,2)=NAME(11,2)=NAME(12,2)=
     #10H
C
C
C     SET UP NUMBER OF FRAMES TIME AVERAGING IS PERFORMED OVER
C
      IF (NG.EQ.0) FRAMES=135.
      IF (NG.GT.0) FRAMES=60.
C
      DO 10 I=1,NFRMS
10    TIME(I)=FLOAT(I)/30.
C
      CALL SCALE(TIME,7.,NFRMS,1)
C
      CALL NEWPEN(1)
      CALL PLOT(0.,-.5,-3)
      CALL PLOT(1.6,2.25,-3)
      CALL POFF
      PRINT *,  " INPUT '-1.' TO REPLOT A PLOT."
      PRINT *,  " INPUT '10.' TO AVOID PLOTTING THE MEAN +/- SIGMA."
      PRINT *,  " "
      READ(5,*) GARBAGE
12    PRINT *, " INPUT DATA SET."
      READ(5,*) SET
      PRINT *, "INPUT PLOT TYPE :1...RMS   2....+-ERROR"
      READ(5,*) ITYPE
      CALL PON
C
      NUM=0
      REWIND 8
      READ(8,22)
15    CONTINUE
```

```
      NUM=NUM+1
      GARB=ABS(GARBAGE)
C
      IF (NUM.NE.9) GO TO 16
      REWIND 8
      READ(8,22)
22    FORMAT(50X,/,50X)
16    CONTINUE
C
      RANGE(1)=RANGE(2)=0.0
C
      MEANAV=MPEAK=SIGAV=0.
      DO 20 I=1,NFRMS
      IF (NUM.LE.8) READ(8,300) EST(I),SIG(I),PP(I)
      IF (NUM.GT.8) READ(8,400) EST(I),SIG(I),PP(I)
      IF (EOF(8).NE.0) GO TO 999
      ESTMS(I)=EST(I)-SIG(I)
      ESTPS(I)=EST(I)+SIG(I)
      RMS(I)=SQRT(SIG(I)**2+EST(I)**2)
      PM(I)=-PP(I)
C
      IF (NG.EQ.0 .AND. I.LE.15) GO TO 19
      IF (NG.GT.0 .AND. I.LE.90) GO TO 19
      MEANAV=MEANAV+EST(I)/FRAMES
      SIGAV=SIGAV+SIG(I)/FRAMES
      IF (ABS(EST(I)).GT.ABS(MPEAK)) MPEAK=EST(I)
C
19    CONTINUE
C
      IF (ESTMS(I).LT.RANGE(1).AND.ITYPE.EQ.2) RANGE(1)=ESTMS(I)
      IF (RMS(I).LT.RANGE(1).AND.ITYPE.EQ.1) RANGE(1)=RMS(I)
C
      IF (PP(I).GT.RANGE(2).AND.ITYPE.EQ.1) RANGE(2)=PP(I)
      IF (RMS(I).GT.RANGE(2).AND.ITYPE.EQ.1) RANGE(2)=RMS(I)
      IF (ESTPS(I).GT.RANGE(2).AND.ITYPE.EQ.2) RANGE(2)=ESTPS(I)
20    CONTINUE
C
      CALL POFF
      IF (NUM.EQ.1) WRITE(6,345)
345   FORMAT(T2,*SET*,T15,*PEAK ERROR*,T30,*AVG ERROR*,
     #T45,*AVG SIGMA*,/)
      WRITE(6,346) NUM,MPEAK,MEANAV,SIGAV
346   FORMAT(T2,I3,T13,3G15.7)
      IF (FLOAT(NUM).LT.SET) GO TO 15
      PRINT *,"INPUT A '1' TO CONTINUE PLOT"
      READ(5,*) GARBAGE
      IF (GARBAGE.NE.1.) GO TO 12
      CALL PON
C
      CALL SCALE(RANGE,4.,2,1)
      HGHT=ABS(RANGE(3)/RANGE(4))
C
      DO 25 I=1,2
      EST(NFRMS+I)=RANGE(2+I)
      ESTMS(NFRMS+I)=RANGE(2+I)
      ESTPS(NFRMS+I)=RANGE(2+I)
      RMS(NFRMS+I)=RANGE(2+I)
      PP(NFRMS+I)=RANGE(2+I)
      PM(NFRMS+I)=RANGE(2+I)
25    CONTINUE
C
40    CONTINUE
C
      CALL AXIS(0.,0.,10HTIME (SEC),-10,7.,0.,TIME(NFRMS+1),TIME(NFRMS
     #         +2))
      IF (NUM.EQ.3.OR.NUM.EQ.4.OR.NUM.EQ.11.OR.NUM.EQ.12) GO TO 50
      IF (NUM.EQ.5.OR.NUM.EQ.6.OR.NUM.EQ.13.OR.NUM.EQ.14) GO TO 60
      CALL AXIS(0.,0.,14HERROR (PIXELS),14,4.,90.,RANGE(3),
     #RANGE(4))
      GO TO 70
```

```
50    CALL AXIS(0.,0.,18HERROR (PIXELS/SEC),18,4.,90.,
     #RANGE(3),RANGE(4))
      GO TO 70
60    CALL AXIS(0.,0.,22HERROR (PIXELS/SEC/SEC),22,4.,90.,
     #RANGE(3),RANGE(4))
70    CONTINUE
C
      CALL PLOT(-1.,-1.,3)
      CALL PLOT(8.,-1.,2)
      CALL PLOT(8.,5.5,2)
      CALL PLOT(-1.,5.5,2)
      CALL PLOT(-1.,-1.,2)
      CALL PLOT(8.,-1.,2)
      CALL PLOT(8.,5.5,2)
      CALL PLOT(-1.,5.5,2)
      CALL PLOT(-1.,-1.,2)
C
      CALL PLOT(0.,HGHT,3)
      CALL PLOT(7.,HGHT,2)
C
      CALL NEWPEN(2)
      IF (ITYPE.EQ.1) GO TO 90
      CALL LINE(TIME,EST,NFRMS,1,15,3)
      CALL SYMBOL(.5,4.2,.1,13H*  MEAN ERROR,0.,13)
      IF (GARB.EQ.10.) GO TO 90
      CALL NEWPEN(3)
      CALL LINE(TIME,ESTMS,NFRMS,1,15,4)
      CALL LINE(TIME,ESTPS,NFRMS,1,15,4)
      CALL SYMBOL(.5,4.0,.1,17H+  MEAN +/- SIGMA,0.,17)
C
      GO TO 110
90    CALL LINE(TIME,RMS,NFRMS,1,15,4)
      CALL SYMBOL(.5,4.2,.1,12H+  RMS ERROR,0.,12)
      CALL NEWPEN(3)
      CALL LINE(TIME,PP,NFRMS,1,15,3)
      CALL SYMBOL(.5,4.0,.1,15H*  FILTER SIGMA,0.,15)
C     CALL LINE(TIME,PM,NFRMS,1,10,3)
C
110   CONTINUE
C
      CALL NEWPEN(1)
      CALL SYMBOL(.2,5.1,.2,16HFILTER ERROR OF ,0.,16)
      CALL SYMBOL(3.8,5.1,.2,NAME(NUM,1),0.,10)
      CALL SYMBOL(6.06,5.1,.2,NAME(NUM,2),0.,10)
C
      CALL SYMBOL(1.0,4.75,.1,6HNRUNS=,0.,6)
      CALL SYMBOL(1.0,4.55,.1,6H   NG=,0.,6)
      CALL SYMBOL(3.0,4.75,.1,6HITARG=,0.,6)
      CALL SYMBOL(3.0,4.55,.1,6HALPHA=,0.,6)
      CALL SYMBOL(5.0,4.75,.1,6HVARDF=,0.,6)
      CALL SYMBOL(5.0,4.55,.1,6H VARM=,0.,6)
C
      CALL NUMBER(1.7,4.75,.1,FLOAT(NRUNS),0.,-1)
      CALL NUMBER(1.7,4.55,.1,FLOAT(NG),0.,-1)
      CALL NUMBER(3.7,4.75,.1,FLOAT(ITARG),0.,-1)
      CALL NUMBER(3.7,4.55,.1,ALPHA,0.,1)
      CALL NUMBER(5.7,4.75,.1,VARDFG,0.,1)
      CALL NUMBER(5.7,4.55,.1,VARM,0.,1)
C
      CALL POFF
      GO TO 12
      READ(5,*) GARBAGE
      CALL PON
      CALL NEWPEN(1)
C
      IF (GARBAGE.LT.0.) GO TO 40
C
      GO TO 15
999   CALL PLOTE(N)
      STOP
      END
-END OF INFORMATION-
```

Vita

Mark R. Kozemchak was born on October 31, 1958 in Latrobe, Pennsylvania. He graduated from Derry Area High School in June 1977 with an Air Force Reserve Officers Training Corps scholarship to attend The Pennsylvania State University. He graduated from The Pennsylvania State University as a distinguished graduate, in May 1981, with a Bachelor of Science degree in Electrical Engineering conferred with highest honors. Second Lieutenant Kozemchak was selected to attend the Air Force Institute of Technology from June 1981 to December 1982 to pursue a Master of Science Degree in Electrical Engineering (Electro-Optics). Lt. Kozemchak is a member of the Institute of Electrical and Electronics Engineers.

Permanent address: 111 Meadow Drive
Latrobe, Pa. 15650

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS
BEFORE COMPLETING FORM

1. REPORT NUMBER
AFIT/GEO/EE/82D-4

2. GOVT ACCESSION NO.
AD-A124 78

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
ENHANCED IMAGE TRACKING:
ANALYSIS OF TWO ACCELERATION MODELS IN
TRACKING MULTIPLE HOT-SPOT IMAGES

5. TYPE OF REPORT & PERIOD COVERED
MS Thesis

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Mark R. Kozemchak
2d Lt, USAF

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Air Force Institute of Technology (AFIT/EN)
Wright-Patterson AFB,Ohio 45433

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Air Force Weapons Laboratory/ARAA
Kirtland AFB NM 87117

12. REPORT DATE
December 1982

13. NUMBER OF PAGES
528

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Approved for public release: IAW AFR 190-17.
Lynn Wolaver
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (ATC)
Wright-Patterson AFB OH 45433

4 JAN 1983

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Extended Kalman Filter
Forward Looking Infrared (FLIR) sensor
First order Gauss-Markov acceleration model
Constant turn-rate acceleration model
2D Fourier Transforms
Atmospheric jitter
Image tracking

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Two extended Kalman filter algorithms that estimate target position, velocity, and acceleration, as well as atmospheric jitter are developed for use within a laser weapon system. Digital signal processing techniques are employed on data obtained from a forward looking infrared (FLIR) sensor in order to identify the underlying shape of "multiple hot-spot" targets. No a priori ingormantion is assumed about such images. The identified shape is used in the measurement model portion of the extended Kalman filters in order to estimate target position offsets from the center of the sensor field

20. ABSTRACT

of view. The two dynamics models incorporated within the filters are 1) a first order Gauss-Markov acceleration model and 2) a constant turn-rate acceleration model. Performance of these two filters is compared in tracking scenarios involving constant G and constant roll-rate maneuvers. Extensive consideration is given to simulating realistic multiple hot-spot images on the FLIR image plane. Performance of a previously developed adaptive filter is shown to be seriously degraded when faced with multiple hot-spot images since it assumes a priori information about the target image. All evaluations are conducted using Monte Carlo techniques.

END

FILMED

3-83

DTIC